TÉCNICAS DE GESTIÓN DE VUELO AUTÓNOMO SOBRE CUADRICÓPTERO

TRABAJO DE FIN DE GRADO
UNIVERSIDAD CARLOS III DE MADRID
CAMPUS DE COLMENAREJO

Sergio Iriz Ricote

Tutor

Jesús García Herrero

26 DE SEPTIEMBRE DE 2016

QUIERO AGRADECER A MIS PADRES Y A MI HERMANO EL APOYO RECIBIDO DURANTE TODA MI ENSEÑANZA, QUE SIN ELLOS NO HABRÍA SIDO POSIBLE LLEGAR HASTA AQUÍ, POR LO QUE QUIERO DEDICARLES EL PRESENTE PROYECTO.

EN SEGUNDO LUGAR, QUERÍA MENCIONAR A MIS COMPAÑEROS DE LOS CUALES ME LLEVO AMIGOS PARA TODA LA VIDA, DE LOS QUE HE APRENDIDO MUCHO Y CON LOS QUE HE CRECIDO Y MADURADO.

COMO OLVIDAR A MIS AMIGOS DE TODA LA VIDA Y LAS PERSONAS ESPECIALES PARA MÍ, QUE ME HAN ACOMPAÑADO EN ESTE IMPORTANTE VIAJE

POR ÚLTIMO, QUIERO AGRADECER A LOS PROFESORES DE LA UC3M SU DEDICACIÓN Y ENSEÑANZA, Y EN ESPECIAL A MI TUTOR DEL PROYECTO Y PROFESOR EN VARIAS ASIGNATURAS DE LA CARRERA JESÚS GARCÍA HERRERO DEL QUE HE APRENDIDO MUCHO Y ME HA APOYADO Y GUIADO EN TODO MOMENTO PARA LA REALIZACIÓN DE ESTE PROYECTO.

"AD AUGUSTA PER ANGUSTA" LA GLORIA SE CONSIGUE LUCHANDO

Sergio Iriz Ricote

Tabla de Contenidos

ÍNDICE DE DIAGRAMAS DEL DOCUMENTO

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

1. INTRODUCTION

The Unmaned Aerial Vehicles, (onwards UAV from the traduction into English), are popularly known than drones, nowadays have become convert into great exploited area with a big request.

A technology, that are generating a great and new business opportunity in the recent years and this are causing an expert great interest.

The development of this technology are including fast in our society on issues such as the professionalization or find a system rules guaranteeing the security and population protection, they consider simultaneously to the application design and new services.

The reason are the possibilities of this new tool are many and varied, could be of a great utility in variety ways as the agriculture, the management of natural disaster among other.

The investigation in the control field of the UAVs was been since the aviation beginnings, the principal reason for the military industry for their different applications in war. But nevertheless, the civil industry application of this technology are significantly growing in recent years thanks to the new research team's appearance including this technology as chief study.

Ilustración 1: UAV en conflicto bélico

This phenomenon is caused by the reduction of costs of the devices, as increase of the power and size relationship offered by the new manufacturing technologies and causes both economy and hardware constraints.

These UAVs began to be manufactured and used during the Second World War for complete dangerous missions or arrive to difficult places for piloted planes. Whit these weapons of war the army could look out air attacks without being discovered.

The obvious advantages as both weight and great mobility caused the fast spread to civilian areas as traffic control, monitoring of disastrous areas, mapping and photography. But, the peak began a decade ago, since the reducing job costs geolocation systems as the GPS. (Díaz, 2016)

The implementation, development and study of the control systems for the UAVs aren't trivial, because have a great variability levels, many constraints in its components means that the classical, linear and mono technical variables are limited to generate an efficient, reliable and robust system control. Control of UAVs requires use advanced techniques to allow have the characteristics and system constraints unmanned and the control variables that we can't control for its component of randomness, typical of any model. In this order of things would be appropriate to use modern modeling techniques and nonlinear control which to achieve an efficient, reliable and robust control in different flight conditions that may be such as the autonomous flight, stability control, speed control, point to point, landing and takeoff flight.

Because of this, one of the booming areas of research in this decade, is being the control System study that allow us to develop a reliable and refined path autonomously control System.

1.1 MOTIVATION PROJECT

Drones (UAVs) are a sector in full swing. Rescues, construction, people tracking, ground survey, recognition, or precision agriculture are some of the ways that performed today without mention the military section.

In addition, it is expected that in future UAVs can be use as satellites, using solar energy, and border surveillance.

This means that currently the drones are assuming a real revolution in the market, as a product and for its incorporation chances to different production processes, or as substitute for previous systems especially for enterprising.

Begin to emerge with increasing strong, professionals and companies involved to use the drones for a multitude of business tasks. Drones are also joining to the agronomist sector, where could be programmed to detect the crops states, or practicing precision agriculture.

This implies that a large number of possibilities exist even in the refinement of the techniques of control and autonomy of drones to discover and define. So many lines of research are coming together to perform a control system that allows an autonomous flight that satisfies more complex tasks that currently many UAVs are capable of performing such as acquiring a UAV that has the ability to fly autonomously based on tracking GPS points, however the particular interest on these aircraft motivates complement these features and enhance them with new tasks that convert these air vehicles in a real efficient system, because as mentioned earlier many simple tasks such as landing or takeoff, calculate a route to a certain point, are severely limited by a trading system based GPS does not take into account a number of possible contingencies during the path to follow, as can be the wind, or an initial deviation due to the orientation of the drone, the inability to react to changes in the environment, or dependency on a user control.

Therefore, the application of modern control techniques and identification of models, which are a challenge from the point of view of control engineering is required.

This has been my main motivation for framing my project in the field of unmanned aerial vehicles . This project is aimed at understanding the structure and operation of a dron of cuadricopter type , getting access its various utilities , for a refining system control, which offers guided the user to be able to perform autonomously system with a path and provide it with the ability to correct the path , starting from different types of positioning either obtained by GPS, or by different sensors cuadricóptero as well as the other information that we can collect the drone , which offset the error given by the system sensors as both path error .

Ilustración 2:Motivación del proyecto

1.2 OBJECTIVES

1.2.1 GENERAL OBJECTIVES

The main objective of the development of this end of grade project is the control system implantation for a system of an autonomous flight of an unmanned aerial vehicle ARDrone 2, which is capable of perform a real time corrections during a flight, reaching a target point of a coordinate axis accurately, minimizing the error and the path deviation.

To achieve this goal, we must achieve a number of objectives:

1. Development of a recognition system of point to point path and drone flying to these points. To do this job we must to study the libraries that we have and the work environment with them, such as the resources and possible moves that offers us the drone.

2. Extracting real time data during the flight Drone, to be treated and used in the design most appropriate control system for flight autopilot. To this it should be investigated which data are most relevant and how we can obtain either from sensors or from a series of mathematical operations.

3. Research and incorporation of different communication protocols between the air base and drone for traffic drone flight information.

4. Processing of data obtained during different flight paths, for a next design of a control system that corrects the path and movement during the flight of the drone.

5. The implementation and integration of the control system in the ARDrone 2 software.

6. Study and incorporation of alternative control software drone outside the flight as well as extraction and processing data to select the best alternative control.

7. Obtaining results of the control system developed through an analysis of the paths made with this control system, and the observation of accuracy and error reduction in conducting paths.

8. Final conclusions and possible applications and tools that we can incorporate into our control system.

1.3 DOCUMENT STRUCTURE

Next will be explained the content and structure in which the document is divided. This structure consists of eleven sections, which are detailed below:

1. Chapter One. Introduction. In this chapter are explain the abstract of the documents, its main objectives and the motivation why the project has been completed.

2. Chapter two. Approach to the problem. In this chapter is explain what are the next steps for the correct development of the project.

3. Chapter three. Theoretical framework. First of all are explained the principal concepts of this project necessary for the understanding thereof, than the drone concept and the classification either by structure or utility drone, the cuadricopter functioning, the nowadays reference systems with their evolution and finally the regulatory framework for this project.

4. Chapter four. Technology used in this Project. Are explained the mains technologies that had needed for the development of this work, such as the hardware drone used, the structure, the sensors and peripherals and the possible flight modes of the drone. Finally, is explained the base station used.

5. Chapter five. Communication protocols. For this Project are very important know how the drone is connected to the base station, and the possibilities that this offer. In this section are explains all this possibilities and connection modes between the drone and the base station, also is informed about a new communication protocol that has been investigated and implemented in this project.

6. Chapter six. LSI Drone Interface. This section explains the needed of use an interface for communicate with the drone, and develop a good control system. First of all is explained interfaces of manufacturing which there are now, and how has been developed the final interface for our system. Also are explained the software employed during the development.

7. Chapter seven. Implementation of the technical solution. In this section are detailed the design and subsequent implementation of the project.

8. Chapter eight. Final diagram for explain the solution and functioning of the system.

9. Chapter nine. Results and evaluation of the system.

10. Chapter ten. Conclusions. Are explained the objectives achieved, the conclusions obtained of the development of the control System and future projects.

11. Chapter eleven. Project planning. Both initial planning and real planning.

2. PLANTEAMIENTO DEL PROBLEMA

En este apartado del documento, se va a realizar una explicación de la puesta en marcha del proyecto que se ha desarrollado, así como las diferentes fases que se han seguido de forma general para llegar al desarrollo del proyecto.

En primer lugar, deberemos conocer las estructuras principales empleadas en la actualidad en los sistemas de vuelo autónomo, en este caso los drones o UAVs. Debemos saber cómo funcionas con sus diferentes estructuras y clasificaciones, así como específicamente comprender como funciona el tipo de Dron con el que vamos a trabajar.

A continuación, al tratarse de un proyecto para el que vamos a desarrollar un sistema de control de vuelo autónomo es imprescindible conocer cuáles son los sistemas por los que se puede guiar el dron, es decir los sistemas de posicionamiento empleados en la actualidad, para seleccionar aquel que sea el más efectivo para nuestra finalidad.

Una vez recabada esta información es imprescindible comprender los sistemas de vuelo actuales no solo en los UAV puesto que estos sistemas emplean el mismo sistema de navegación que cualquier vehículo aéreo. Debemos conocer sus diferentes trayectorias, así como los modos de vuelo que pueden entrar en funcionamiento en cada momento.

Finalmente debemos estudiar la tecnología que vamos a utilizar en el desarrollo del proyecto, el dron empleado con su hardware y ensamblado, como el software que empleamos y desarrollamos para su control. Es importante también diferenciar las diferentes partes que entran en juego en la comunicación con el dron para un sistema de navegación autónomo, como son los elementos que interactúan en la navegación, como los diferentes protocolos de comunicación que han sido empleados.

Una vez cumplido la primera parte de familiarización con el entorno y contexto del proyecto, así como la investigación de las diferentes alternativas y posibilidades que se han introducido en nuestro desarrollo, debemos avanzar hacia la siguiente fase del proyecto.

La segunda fase se trata del diseño del sistema de control que vamos a emplear realizando un sistema de navegación inteligente que sea capaz de corregir su trayectoria de forma autónoma y llegar a sus objetivos de forma precisa para realizar el desarrollo de la solución técnica a este diseño.

En tercer lugar, llegaremos a la fase de implementación de la solución técnica previamente diseñada, en la plataforma de programación y el dron. Llegando en último lugar a una fase de análisis de los resultados obtenidos.

3. MARCO TEÓRICO

3.1 DRONES Y CUADRICÓPTERO

3.1.1 ¿QUÉ ES UN DRON?

Por definición un dron, también conocido como vehículo aéreo no tripulado UAV, es una aeronave que vuela sin necesidad de ser controlado por un humano en su interior. Un UAV se define como un vehículo sin tripulación reutilizable, capaz de mantener un nivel de vuelo controlado y sostenido, y propulsado por un motor de explosión o de reacción dependiendo de la tipología del dron[1] (Criado, Diseño de estrategias de control para el seguimiento de trayectorias de un cuadricóptero comercial, 2014)

Ilustración 3: Imagen de diferentes drones

Como vehículo aéreo puede tener diferentes formas, que serán estudiadas a continuación. Pero los drones no son algo nuevo, el ejemplo más antiguo fue desarrollado después de la primera guerra mundial, y se emplearon durante la segunda guerra mundial para entrenar a los operarios de los cañones antiaéreos. Sin embargo, no es hasta poco más que a finales del siglo anterior cuando comienzan a utilizarse los drones mediante radio control con todas las características de autonomía. (Hedrich, 2015)

Algunos tienen sistema GPS que les permite reconocer los puntos que ha recorrido y donde se encuentra. En la actualidad y principal motivo de este proyecto se está estudiando con mucho énfasis en conseguir la autonomía de dron, es decir la

capacidad de ir tomando sus propias decisiones, evitando chocar contra las personas y poder evitar los objetos o simplemente llegar a un punto deseado. En este sentido se han desarrollado dos variantes, como son el control desde una ubicación remota, y el vuelo autónomo sobre una base de planes de vuelo usando sistemas más complejos de automatización (Rizo, 2014).

Cabe destacar que las aeronaves controladas remotamente en realidad no califican para ser llamadas como UAV, ya que los vehículos aéreos pilotados remotamente se conocen como Aeronaves R/C; esto debido a que, precisamente, los UAV son también sistemas autónomos que pueden operar sin intervención humana alguna durante su funcionamiento en la misión a la que se haya encomendado, es decir, pueden despegar, volar y aterrizar automáticamente.

Se utilizan para múltiples tareas, desde tareas de vigilancia, fotografía, retransmisiones televisivas, agricultura, ocio y muchas más tareas, ya que cada poco se descubre una nueva forma de utilizar los drones (Gómez, 2016).

3.1.2 CLASIFICACIÓN DE LOS DRONES

En la actualidad existen un gran número y variedad de tipos de drones según diferentes criterios de clasificación, ya sea por su tipo de uso o utilidad y por las características físicas.

3.1.2.1 POR SU UTILIDAD

El siguiente gráfico muestra las diferentes utilizaciones en la actualidad de los drones y por lo tanto su distinción o clasificación:

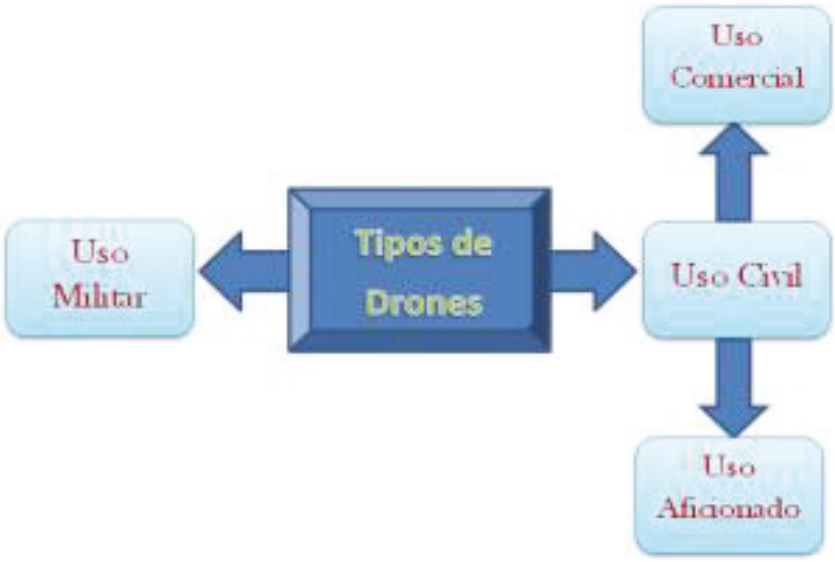


Diagrama 1:Clasificación por utilidad del dron

1) Uso Militar: Las siglas empleadas son UCAV, “Aviones no Tripulados de Combate”, son los mayores impulsores de este tipo de vehículos. Empleados en tareas como:

   a) Blanco. Simulaciones de ataques sobre naves enemigas.

   b) Reconocimiento de zonas de difícil acceso o zonas catastróficas.

   c) Combate, empleados en misiones de alto riesgo

2) Uso Civil: Empleados en tareas de logística, naves de transporte y desplazamiento de cargas. Investigación y desarrollo, como es el caso de este proyecto. Seguridad como puede ser reconocimiento de zonas catastróficas o luchas contra incendios. Ocio, este uso es muy variado desde grabaciones o como juguete (Wikipedia, Wikipedia, 2016).

3.1.2.2 POR SU ESTRUCTURA Y CARACTERÍSTICAS

Nos encontramos ante dos tipos de drones según su estructura: Ala fija y Ala rotatoria.

El ala fija tiene el aspecto y forma de un avión de aeromodelismo de toda la vida y el ala rotatoria tiene dos tipos de drones incluidos, los helicópteros y los multirrotores.

3.1.2.2.1 ALA FIJA

El Dron de ala fija es el tipo de dron con una mayor autonomía. Está equipado con un motor eléctrico o de explosión, puede permanecer en el aire más tiempo. Además, es el dron con una mayor capacidad aerodinámica ya que si le dotamos de una configuración adecuada, puede permanecer bastante tiempo sin necesidad de utilizar el motor gracias al planeo. Por otra parte, el hecho de poder planear hace que sea una plataforma mucho más segura, ya que en un supuesto fallo de motor puede planear hasta llegar al punto de aterrizaje (Area Tecnología, 2015).

Ilustración 4:drones de ala fija

Sin embargo, el ala fija, está preparado para unos fines muy específicos, lo que le resta versatilidad a la hora de ser utilizado. Su principal desventaja es el tema del aterrizaje y el despegue.

3.1.2.2.2 ALA ROTATORIA

3.1.2.2.2.1 HELICÓPTEROS

Poseen una gran capacidad de autonomía. Debido a que posee un solo motor y una hélice de gran tamaño. El helicóptero es mucho más eficiente aerodinámicamente que un multirrotor, ya que el helicóptero funciona a revoluciones fijas de motor gracias al movimiento variable de las hélices, mientras que el multirrotor varía las revoluciones del motor para mantenerse estable (Pose, 2015).

Ilustración 5:Helicóptero

Sin embargo, los helicópteros son bastante complejos a nivel mecánico, lo que nos obliga a tener que estar constantemente ajustándolo para que nos ofrezca un vuelo óptimo. Por este motivo la mayoría de los UAV empleados son multirrotores, debido a su mayor sencillez y estabilidad.

3.1.2.2.2.2 MULTIRROTORES

Este tipo de drones son la herramienta más extendida la más empleada por ello. Proporciona una gran versatilidad y eficacia en las operaciones por su simpleza a la hora de ser pilotados y por la velocidad de montaje. Es una plataforma estable por naturaleza, debido a que los motores se encuentran a la misma distancia del centro de gravedad de la aeronave.

- Según la cantidad de motores de los que disponga se realiza una clasificación en:

1) **Tricópteros**. Este tipo de drones se componen de tres motores, tres reguladores, un servomotor y cuatro hélices. Los tres motores se localizan al final de los tres brazos. En ese mismo lugar, podemos encontrar además un sensor. Las hélices dirigen a los reguladores, los cuales se encuentran en el tronco del objeto. El motor para la estación emisora es simplemente un servomotor normal en una única dirección.

2) **Cuadricópteros o quadrotor**. Un cuadricóptero es un dron multirrotor con cuatro brazos, los cuales tienen en su parte final un motor y una hélice. Son parecidos a los helicópteros en muchos aspectos, aunque la elevación y el empuje lo realizan con cuatro hélices en vez de una. El dron empleado para el desarrollo de este proyecto es de esta tipología.

3) **Hexacópteros y Octocópteros**. Dron conformado por 6 y 8 brazos respectivamente con un motor en cada extremo, con mayor carga y potencia que los drones anteriores. Normalmente empleados en uso militar.

Además, podemos realizar una distinción de la anterior tipología de drones dependiendo de la configuración de los brazos ya sea en **“Y”,” X”, “Y invertida” o “+ (Cruz)”**.

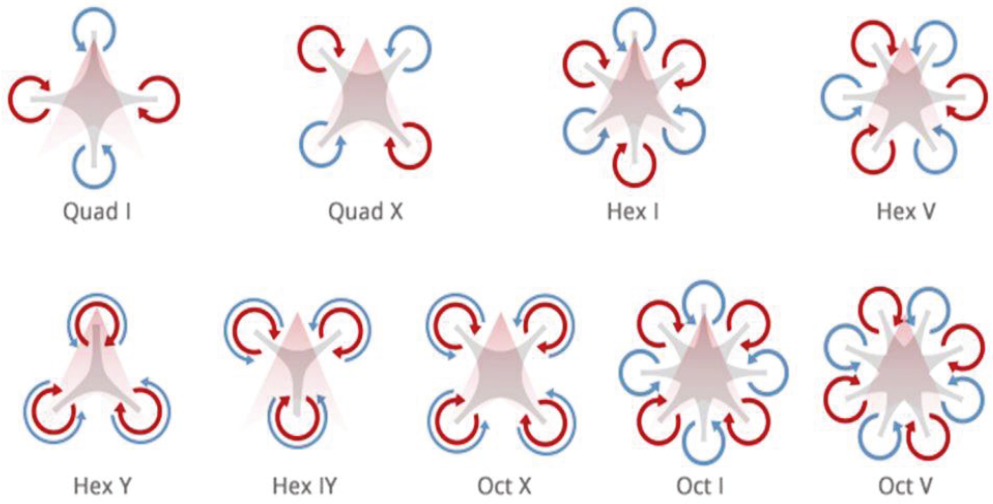


Ilustración 6: Tipos de multirrotor

Por otro lado, están los multirrotores coaxiales, es decir, dos motores por brazo. Se caracterizan por un menor peso, pero una menor aerodinámica.

Una característica a tener en cuenta es que a mayor número de brazos tendremos más estabilidad y más seguridad, mientras que cuantos más motores tengamos más propulsión y consumo. (Pose, 2015)

3.1.2.2.2.2.1 VENTAJAS

El motivo de elegir un multirrotor para el desarrollo de este proyecto principalmente ha sido por las siguientes ventajas:

1) **Modularidad**: Los componentes del sistema tienen la posibilidad de reemplazarse por otros diferentes. Un caso muy común es el cambio de los conjuntos motor-hélice para lograr equipos del estilo acrobático o de larga duración.

2) **Versatilidad**: Permiten la inclusión de diferentes softwares compatibles con diferentes modelos de drones, permitiendo así incluir diferentes dispositivos para ayudarnos a realizar el sistema de control.

3) **Facilidad de manejo**: Al tener un despegue y aterrizaje vertical, además de una gran posibilidad de movimiento y giro, convierten al dron en un vehículo muy adaptativo a cualquier tipo de entornos además de dotarlo de una gran capacidad para maniobrar.

3.1.3 DIAGRAMA DE CLASIFICACIÓN DE DRONES

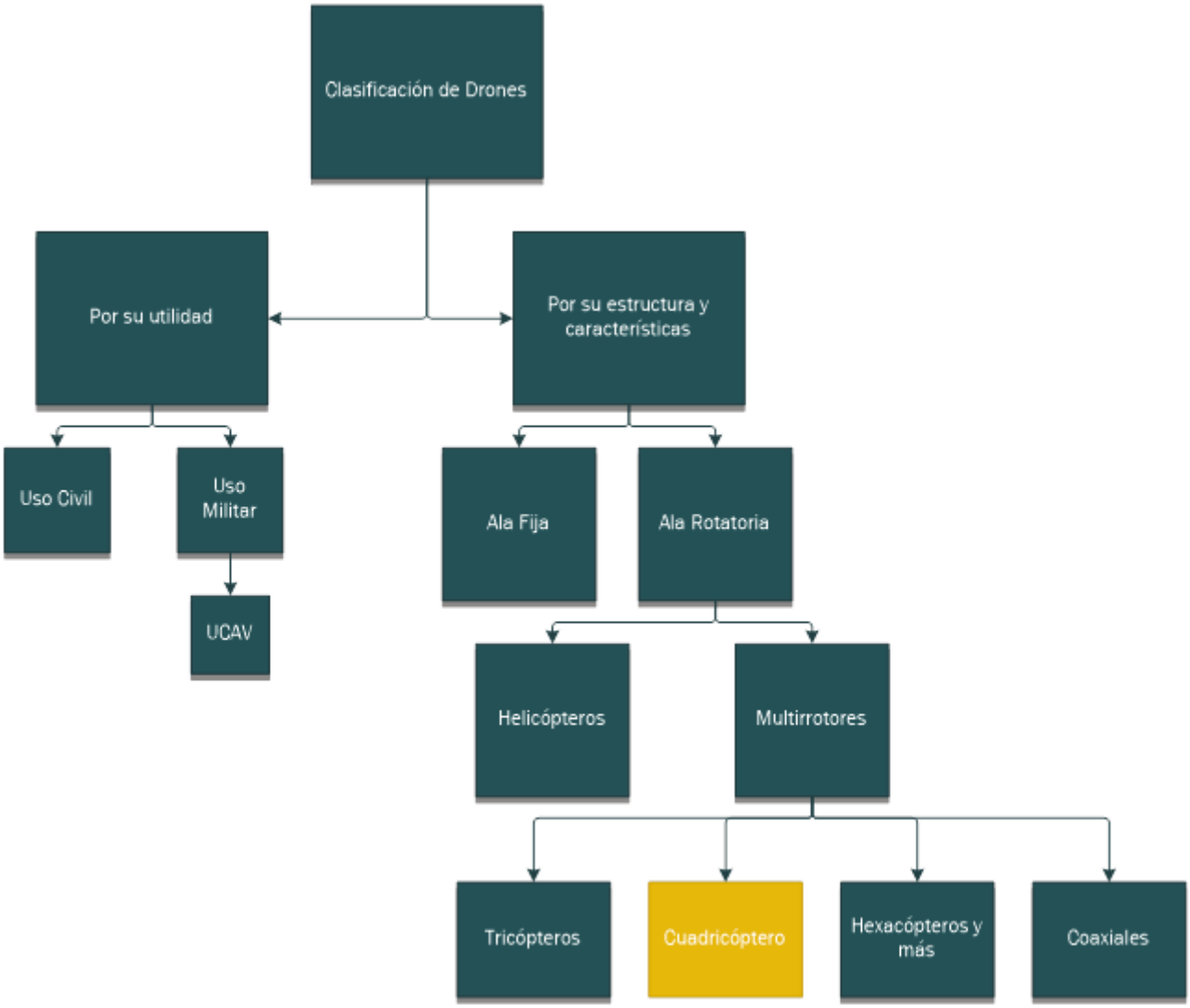


Diagrama 2: Clasificación de drones

3.2 FUNCIONAMIENTO DEL CUADRICÓPTERO

Debido a que para el desarrollo de este proyecto vamos a emplear un Sistema cuadricóptero AR Drone 2, del que es esencial conocer su funcionamiento y su estructuración.

Como ya conocemos el cuadricóptero es una aeronave dotada de cuatro motores, dispuesto en brazos con forma de equis o cruz, con el complejo sistema de medición inercial y control que permite realizar el vuelo estable.

Entendemos por estable a la capacidad del dron para permanecer en una misma posición sin control del usuario que está encargado de controlarlo en el mayor tiempo posible. Este control mencionado, se consigue variando la velocidad angular de cada uno de los sistemas motrices.

La característica principal y más representativa de este tipo de drones es la simetría en su estructura y la rotación asimétrica dos a dos, con la que se consigue un equilibrado del dron en todo momento y una gran variedad de giros, es decir existen cuatro motores que giran en el mismo sentido dos a dos con diferente sentido de rotación, unos en sentido horario y otros anti horario. (Pose, 2015)

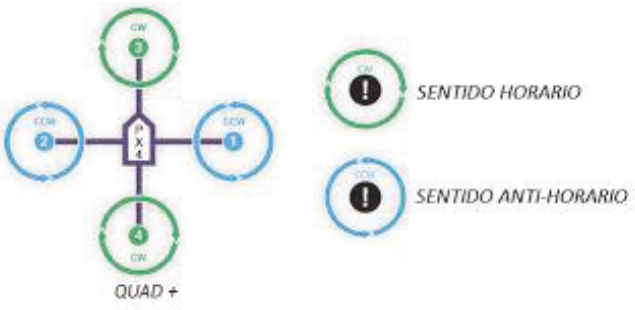


Ilustración 7:Funcionamiento de un cuadricóptero

La diferencia de la distribución de los brazos en cruz o equis radica principalmente en qué se considera como retroceso, avance, izquierda y derecha. La opción en cruz es la más sencilla de comprender y programar ya que los movimientos de cabeceo y alabeo, pitch y roll a partir de ahora haciendo referencia a los términos empleados en aeronáutica, se realizan dejando un par de motores a una revolución permanente o constante y realizando variaciones en el otro par de rotores, de tal modo que uno de ellos aumentará sus revoluciones y otro las bajará.

Anteriormente hemos mencionado dos tipos de giros, pero el sistema general de movimiento de un cuadricóptero y la navegación de aeronaves en general se realiza sobre un sistema de ejes, produciéndose tres tipos de giros.

Este sistema de movimiento se realiza sobre un sistema de coordenadas de un plano XYZ. El eje X o **eje longitudinal**, se trata del eje imaginario que va desde el frontal del dron, hasta su cola y el movimiento que se realiza sobre este eje se denomina "Roll".

El **eje transversal** o lateral hace referencia al eje Y, que se trata de un eje imaginario que va desde el extremo de un brazo del dron a otro, el movimiento sobre este eje es el "Pitch".

Y por último **el eje vertical** o eje Z, que se trata del eje que atraviesa el dron. Sobre este eje se realizará el movimiento denominado guiñada, "Yaw" a partir de ahora.

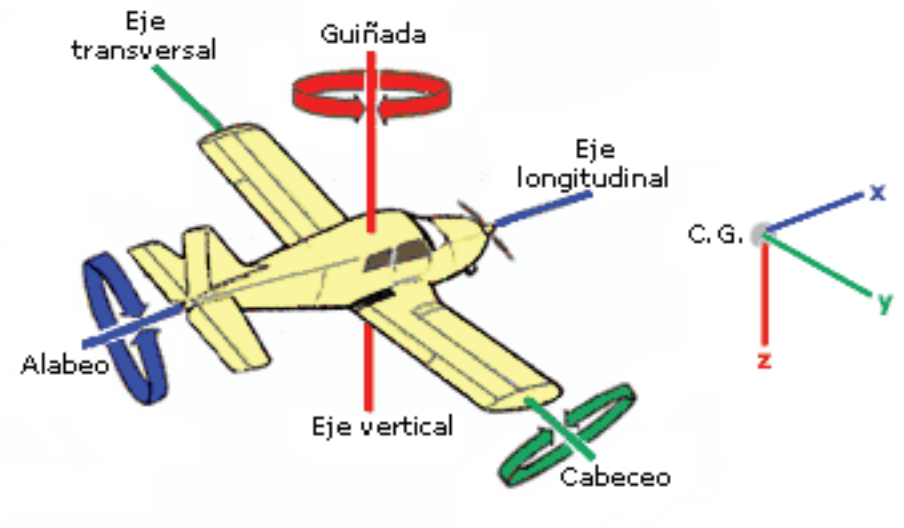


Ilustración 8:Funcionamiento de un cuadricóptero (2)

El origen de coordenadas será el centro del dron siendo el centro de gravedad del dron. (Vicedo, 2013-2014)

Este tipo de movimiento, se calcula empleando unos ángulos de navegación Eulerianos empleados para conocer la posición del dron o la aeronave en un momento dado respecto de los ejes de coordenadas fijos.

Roll (Alabeo): rotación alrededor del eje longitudinal del avión. Este movimiento se obtiene modificando los motores opuestos uno aumentando su velocidad y el otro disminuyéndola con la misma variación.

Pitch (Cabeceo): inclinación del morro del avión, o rotación respecto al eje ala-ala. La obtención de este movimiento o giro es la misma que para el Roll, pero empleando los motores restantes.

Yaw (Guiñada): movimiento del avión respecto del eje imaginario vertical que pasa por el centro de gravedad de la aeronave. Para ajustar el Yaw se sube el empuje de dos motores opuestos mientras los otros dos se mantienen estables

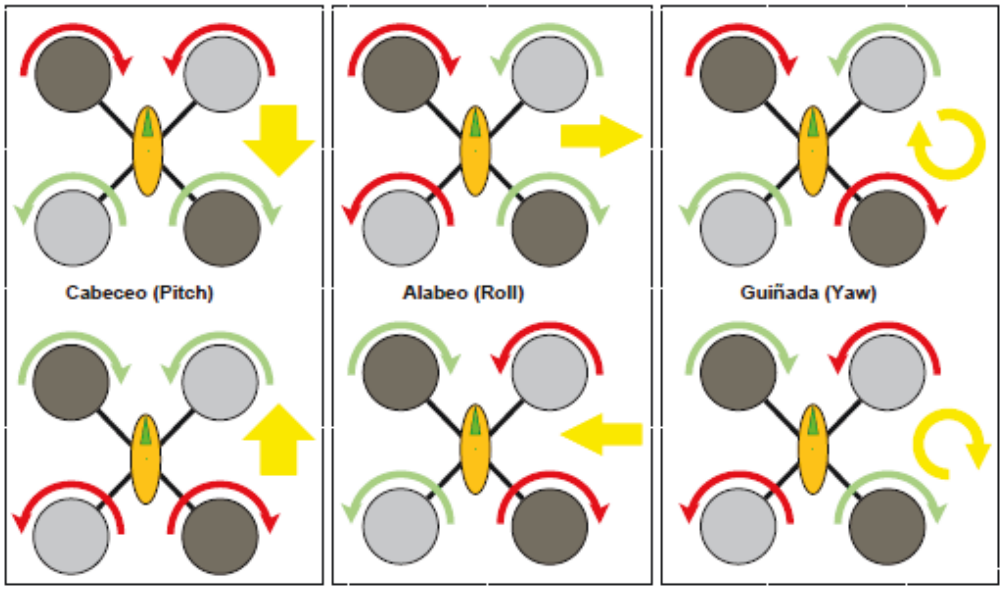


Ilustración 9:Funcionamiento de un cuadricóptero (3)

Ilustración 10:Funcionamiento de un cuadricóptero (4)

Y por último hay que mencionar el movimiento de **altitud**, que se trata de aquel movimiento empleado tanto para despegar como para aterrizar, así como aumentar o disminuir altura. Este tipo de movimiento no emplea giros ni variaciones asimétricas de la velocidad de los rotores, simplemente establece una velocidad igual y constante para todos ellos, y aumenta o disminuye en el caso de querer ganar o perder altura. (Criado,

Diseño de estrategias de control para el seguimiento de trayectorias de un cuadricóptero comercial, 9 diciembre 2014)

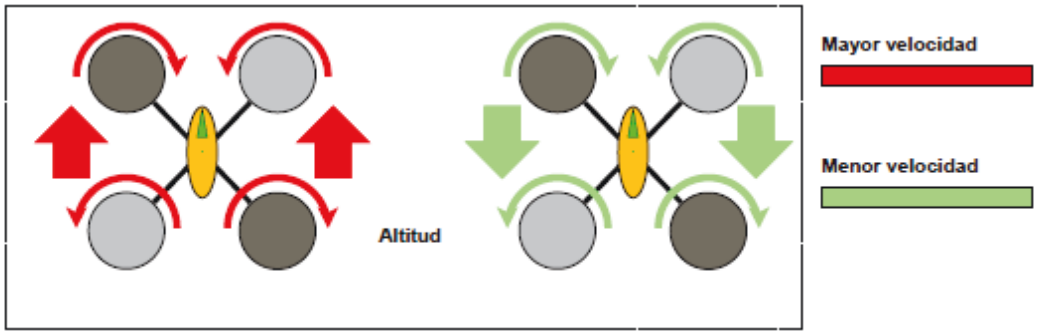


Ilustración 11:Funcionamiento de un cuadricóptero (5)

3.3 SISTEMAS DE POSICIONAMIENTO ACTUALES

Para el desarrollo de un Sistema de vuelo autónomo de un dron, es necesario basarse en un sistema que nos indique cuál es su posición actual, siendo no solo su posición global, sino también su posición relativa respecto a un punto de partida.

3.3.1 SISTEMA DE POSICIONAMIENTO GLOBAL.

GPS hace referencia a las siglas “Global Positioning System” empleando la constelación NAVSTAR (NAVigation System for Time And Ranging) como Sistema de posicionamiento global, cuya funcionalidad principal es determinar la posición de la aeronave a partir de las señales radioeléctricas de un conjunto de 24 satélites.

Esta metodología surgió como medio de mejora del sistema de satélites de uso militar, que se ha extendido en todo el mundo desde 1967.

El sistema de referencia que emplea se conoce como WGS, que se trata de un elipsoide global de referencia nacido en 1960 hasta su versión mejorada WGS84, del que se obtienen las coordenadas cartesianas o polares del punto en el que se encuentra la aeronave. Para obtener dichas coordenadas a través del GPS es necesario saber que deben realizarse una serie de transformaciones matemáticas para obtener los puntos de referencia del sistema de coordenadas local que se precise, ya que las coordenadas están referidas al sistema WGS84. (Farjas, 2016)

Los parámetros de esa transformación son un total de siete, tres traslaciones (Tx, Ty, Tz), tres rotaciones (Rx, Ry, Rz), y un factor de escalado y se obtiene a partir de puntos con coordenadas conocidas en el sistema inicial (WGS-84) y en el sistema final (local).

Dicho sistema local podrá ser un sistema local producido a partir de unas coordenadas aleatorias asignadas a un punto cualquiera de referencia, o pueden emplearse como sistema regulado como el sistema ED-50, empleado en la cartografía oficial española (geográfico, 2016).

3.3.2 SISTEMAS ELIPSOIDALES DE REFERENCIA.

La figura de la Tierra, se asemeja a la definición de geoide, siendo una superficie de nivel equipotencial del campo gravitatorio terrestre. Esta es la superficie empleada como referencia de la altitud.

Debido a la gran complejidad de representación de un geoide se emplea como representación de la Tierra un elipsoide de revolución, por ello el elipsoide de revolución que mejor se adapte al geoide en un punto de referencia donde queremos representar nuestras posiciones o bien la normal a ambos es la solución adoptada, constituyendo el concepto de Sistema Geodésico de Referencia (geográfico, 2016).

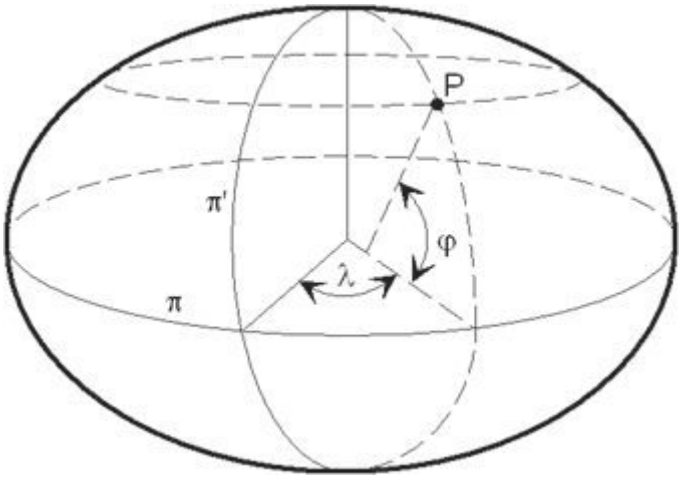


Ilustración 12: Sistema Geodésico de Referencia

Sobre esta superficie se definen las coordenadas geodésicas:

Latitud geográfica (ϕ): ángulo medido sobre el plano meridiano que contiene al punto entre el plano ecuatorial y la normal al elipsoide en P.

Longitud geográfica (λ): ángulo medido sobre el plano ecuatorial entre el meridiano origen y el plano meridiano que pasa por P. (Mínguez, Abril,2009)

3.3.3 SISTEMA WGS84

Como se ha mencionado con anterioridad el GPS emplea el sistema de referencia WGS84 de posiciones y vectores, definiéndose como un sistema cartesiano geocéntrico con origen en las masas de la Tierra, un eje Z paralelo a la dirección del polo, un eje X representado por la intersección del meridiano origen, Greenwich, y el plano que pasa por el origen y es perpendicular al eje Z y un eje Y ortogonal a los anteriores (geográfico, 2016)

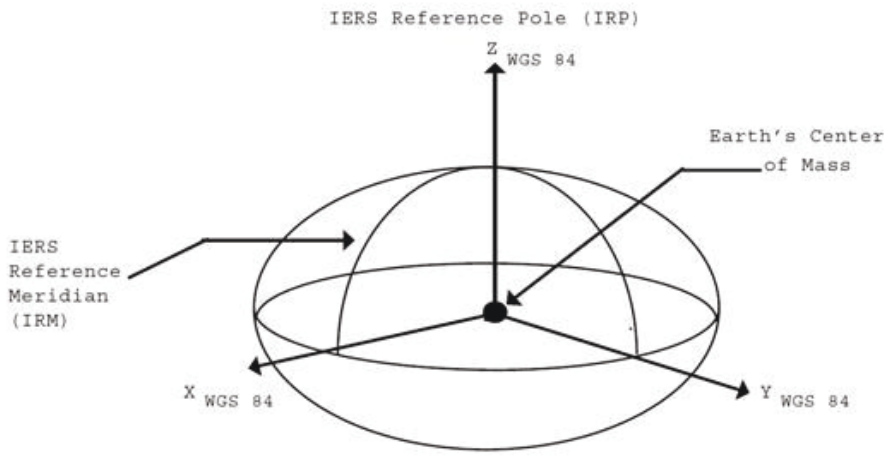


Ilustración 13: Sistema de referencia WGS84

Como podemos observar el sistema de ejes es similar al sistema de navegación (pitch, roll y yaw) empleado en la navegación del cuadricóptero (geográfico, 2016).

3.3.4 SISTEMA DE REFERENCIA LOCAL

Todas las redes geodésicas se calculadas sobre un sistema de referencia que está definido por los siguientes elementos.

1) El elipsoide de referencia.
2) El punto de coincidencia entre el geoide de la Tierra y el elipsoide de revolución.
3) Origen de longitudes y latitudes o de los ejes de coordenadas.
4) Origen de altitudes.

Al conjunto de datos anteriores con el que determinaremos el sistema de posicionamiento local se le conoce como DATUM. (Mínguez, Abril,2009)

3.4 MODOS DE VUELO

En general los drones suelen tener distintos modos de vuelo categorizados dependiendo del grado de autonomía, existe una multitud de modos de vuelo específicos para cada dron, pero los más típicos y principales son tres.

3.4.1 MODO MANUAL.

Es un tipo de control abierto, donde el dron realiza unos cálculos para la velocidad de giro como salida de los motores, a partir de una serie de comandos, explicados en secciones anteriores, como son el pitch, roll y yaw, además del despegue o aterrizaje. Por lo general este modo de vuelo no suele ser empleado en los multirrotores UAV.

3.4.2 MODO ESTABLE O ESTABILIZADO.

El control se hace de forma de lazo cerrado, es decir comienza en un punto y finaliza en el mismo. Se emplea un sistema de control de orientación encargado de calcular la salida de las velocidades de los motores a partir de los controles pitch, roll y yaw que envía el usuario que controla el sistema y de los valores de retroalimentación que genera el dron. La diferencia principal con el modo manual es que a pesar de recibir los controles por parte de un usuario que los realiza de forma manual, el dron es capaz de auto estabilizarse por sí solo en un sistema espacial de tres dimensiones XYZ.

3.4.3 MODO AUTÓNOMO.

El sistema no recibe ninguna instrucción por parte del usuario, sino tan solo una serie de coordenadas en tres dimensiones, ya sean locales o globales, siguiendo una serie de waypoint o puntos de paso que recibe como entrada, siendo capaz de avanzar hacia ellos de forma estable y autónoma.

En los cuadricópteros se suele utilizar o designar como modo manual al modo estable. Además, se suelen implementar distintos modos de vuelo estable, donde alguno de los controles es manejado directamente por el dron, dejando al usuario la posibilidad de controlar sólo ciertas acciones. (Quirós, 2015-02-22)

3.5 MARCO LEGAL DE DRONES EN ESPAÑA

Es muy importante para realizar este proyecto conocer las limitaciones legales que tenemos para el uso de drones en nuestro país para este desarrollo.

De esta forma podremos tener un sistema de control de vuelo autónomo que cumpla la legalidad vigente y podamos volar nuestro dron con total libertad.

3.5.1 LEY 18/2014

El Consejo de Ministros el viernes 4 de julio de 2014 aprobó el régimen temporal para las operaciones con aeronaves pilotadas por control remoto, los llamados drones, de peso inferior a los 150 kg al despegue, en el que se establecen las condiciones de explotación de estas aeronaves para la realización de trabajos técnicos y científicos.

Sin embargo, dicho Real Decreto-ley dio lugar a la tramitación parlamentaria de un proyecto de ley que ha acabado siendo la Ley 18/2014, del 15 de octubre, de aprobación de medidas urgentes para el crecimiento, la competitividad y la eficiencia. ("Ley 18/2014"). La Ley 18/2014 incluye en su art. 50 determinadas disposiciones respecto de las aeronaves civiles pilotadas por control remoto. (Garrido, 2014)

3.5.1.1 INSCRIPCIÓN E IDENTIFICACIÓN

Las UAVs con masa máxima de despegue superior a 25 kg deberán constar en el Registro de Matrícula de Aeronaves y contar con una placa de identificación de la aeronave y su operador. (Ley de regulación de drones en España 18/2014, 2014)

3.5.1.2 DESTINO

Las UAVs podrán destinarse a operaciones especializadas (también conocidas como trabajos aéreos), tales como investigación, fumigación, fotografía, vigilancia, patrulla, publicidad, lucha contra incendios o salvamento, así como a trabajos científicos. (Ley de regulación de drones en España 18/2014, 2014)

3.5.1.3 CONDICIONES DE OPERACIÓN

Dependiendo de la masa de despegue y de si la operación está dentro del campo visual del piloto, se establecen determinados límites de distancias y de altura, así como restricciones para su uso en núcleos urbanos y en las proximidades de aeropuertos y aeródromos. En todo caso, los vuelos deberán realizarse de día y en condiciones meteorológicas visuales. Las operaciones requieren de una determinada documentación, incluyendo un manual de operaciones, un estudio aeronáutico de seguridad, la superación de vuelos de prueba, programas de mantenimiento, medidas de protección frente a interferencias ilícitas y medidas de protección de las personas y los bienes. Estos requisitos se relajan en parte en determinadas condiciones si se trata

de vuelos de prueba, de demostración, de investigación, de desarrollo o de I+D. (Ley de regulación de drones en España 18/2014, 2014)

3.5.1.4 ACREDITACIÓN DE PILOTOS

Los pilotos deben obtener la correspondiente licencia y demostrar conocimientos teóricos, además de cumplir con otra serie de requisitos.

Dependiendo del tipo de UAV, se precisará un tipo de habilitación distinto:

- UAVs con masa de despegue que no exceda de 25 kg: bastará una comunicación previa, junto con una declaración responsable, a presentar con 5 días de antelación al inicio de la operación. La documentación deberá presentarse ante la Agencia Estatal de Seguridad Aérea. (Ley de regulación de drones en España 18/2014, 2014)
- UAVs con una masa de despegue superior a 25 kg: se requiere autorización previa. La autorización debe solicitarse a AESA y, en caso de no obtener resolución expresa, se entenderá desestimada por silencio administrativo. (Ley de regulación de drones en España 18/2014, 2014)
- Se permite la operación de UAVs sin habilitación siempre que se cumplan las condiciones de operación, bajo responsabilidad del operador y cuando así le sea requerido por la autoridad competente. (Ley de regulación de drones en España 18/2014, 2014)

3.5.2 NUEVO REAL DECRETO-LEY

A principios de este año se ha informado acerca de la intención de la modificación de la ley 18/2014, aún no es sabido con total en que momento entrará en vigor, pero se espera que entre en vigor a finales del año 2016. Lo que si sabemos con total seguridad es la modificación de tres aspectos en esta nueva ley:

- Vuelo dentro de núcleos urbanos. Se permitirá el vuelo dentro de núcleos urbanos, ya que la actual ley no lo permite. El piloto deberá regirse a unas normas concretas, como la del vuelo de aeronaves de peso inferior a 10 kg de peso, no alejar el dron a más de 100 metros de distancia, no superar 120 metros de altura y solicitar un permiso previo a una Subdelegación del gobierno. (Dronespain, 2016)
- BVLOS de kg a 5 kg. Con la ley actual no puedes alejar un dron de más de 2 kg a más de 500 metros, lo cual impedía a muchas operadoras hacer ciertos trabajos. Ampliando el peso de la aeronave a 5 kg, ya encontramos en el mercado drones multirrotores y de ala fija capaces de hacer infinidad de trabajos profesionales. (Dronespain, 2016)
- Volar dentro del espacio aéreo controlado. (Dronespain, 2016)

4 TECNOLOGÍAS EMPLEADAS

4.1 ARDRONE

Con este trabajo trataremos de desarrollar un Sistema de control encaminado al vuelo autónomo punto a punto con correcciones en tiempo real sobre un UAV, por ello es muy importante emplear un UAV que nos facilite el objetivo al que se quiere llegar y que además nos permita una gran libertad a la hora de trabajar con su software, disponga de un buen soporte en caso de avería, además de disponer de unas características físicas propicias para realizar un buen control de trayectorias dotándonos de gran libertad de movimiento.

Por este motivo hemos empleado el AR Drone 2.0 de la compañía francesa Parrot, el cual es uno de los drones UAV más vendidos y famosos del mercado. Esto se debe a que el AR Drone 2.0 es un cuadricóptero de bajo coste y con posibilidad para realizar implementaciones de software que en un principio ha sido pensado como un vehículo de ocio que como un dron para realizar tareas más robustas.

Debido a esto, sus características resultan un poco limitadas para determinadas tareas, sin embargo, para fines investigativos y para los alcances específicos de este proyecto presenta una gran ventaja en términos de facilidad de uso y libertad de trabajo con él.

A esto se le suman las mejoras sobre su antecesor el AR Drone incorporando un acelerómetro de tres ejes que nos dota de una gran libertad y movimiento, una serie de sensores para recopilar información de vuelo y la posibilidad de realizar vuelos sin conocer su posición relativa, es decir no necesita conocer la orientación del UAV para realizar un vuelo permitiendo un gran margen y variedad de movimientos en su control.

Otra de sus grandes ventajas es su interfaz de vuelo que, a pesar de no ser empleada en este proyecto, permite realizar vuelos sencillos pilotados y controlados por el usuario para en nuestro caso realizar pruebas de vuelo.

Pero la principal razón de empleo de dicho dron es que tenemos una gran libertad para modificar su software y así mejorar las técnicas de vuelo y control que emplea de serie, pudiendo realizar una gran labor de investigación y de pruebas demostrando que podemos dotar de una gran autonomía a un vehículo aéreo no tripulado.

El AR Drone es capaz de llevar a cabo los dos tipos de navegación principales en los drones, la navegación supervisada por parte de un usuario, o la navegación autónoma empleando un sistema de control como el desarrollado en este proyecto de fin de grado, en ambos casos la navegación se realiza empleando los sensores anteriormente

mencionados tanto de visión como de altitud, efectuando diferentes maniobras de movimiento durante el vuelo a lo largo de los ejes x, y, z, como su altitud.

Para realiza el movimiento lleva cuatro motores, que variando su velocidad como se ha comentado con anterioridad (Funcionamiento del cuadricóptero) consigue llevar a cabo las diferentes maniobras de giro, aterrizaje y despegue.

4.1.1 ESPECIFICACIONES ARDRONE 2

4.1.1.1 HARDWARE DEL CUADRICÓPTERO

4.1.1.1.1 ESQUELETO

Su estructura central está formada por el cuerpo que contiene los elementos principales del Dron. Tiene una batería de litio recargable que permite una autonomía de vuelo de aproximadamente treinta minutos. El sistema informático integrado consiste en un microprocesador ARM9 RISC de 32 bits a 468 MHz, una memoria DDR SDRAM de 128 MB, un sistema operativo con núcleo Linux, por lo que emplea un sistema operativo Open Source, un modem Wi-Fi y además dispone de un conector USB de alta velocidad para poder almacenar los vídeos y las fotos que se realicen.

Podemos decir que tiene una estructura ligera, ya que su peso máximo es de 420 gramos con la carcasa incluida. Eso y el alcance de la conexión Wi-Fi le permiten llegar a altitudes de hasta 50 - 120 metros y alcanzar velocidades de hasta 64 m/s.

La carcasa exterior mencionada es opcional, podemos instalarla o no en el dispositivo si queremos volar tanto en exteriores como en interiores. (Poveda, 2012-2013) (Parrot, ardrone-2, 2016)

4.1.1.1.2 MATERIALES

- El UAV está revestido por tubos de fibra de carbono con peso total de 380 con el casco de protección para el exterior, 420 g con el casco de protección para el interior, que protegen al Dron de posibles impactos y dotan al sistema de una mayor estabilidad.
- Piezas de plástico nylon cargado con 30% de fibra de vidrio de alta calidad para evitar un demasiado peso del vehículo y conseguir una gran ligereza de vuelo.
- Espuma para aislar el centro de inercia de las vibraciones de los motores.

- Casco de protección de polipropileno expandido (PPE) inyectado por un molde metálico.
- Nanorevestimiento repelente a los líquidos en los sensores de ultrasonidos.
- Reparable en su totalidad (Parrot, ardrone-2, 2016)

4.1.1.1.3 CONTROLADOR

El ARDrone 2 contiene una placa controladora sencilla con una arquitectura AR M Cortex A8, procesador OMAP3630 con velocidad de 1 GHz y una RAM DDR de 128MB.

4.1.1.1.4 SENSORES

El dron dispone de varios sensores para capturar y obtener información sensorial del entorno que rodea al vehículo y permitir trasmitirla a la estación base. Los sensores principales están orientados a la visión y a la altitud, careciendo de cualquier tipo de herramienta hardware para obtener datos de sus posiciones. Los dos principales son dos cámaras que capturan imágenes en tiempo real:

Cámara frontal, es la cámara principal del UAV que son empleadas para navegar a través de mapas topológicos definidos previamente. La calidad de dicha cámara es de 720 HD con dimensiones de 640x360 píxeles con color RGB. Se ubica en la parte frontal del UAV que se comunica con el procesador del Dron mediante un cable de comunicaciones. En definitiva, es un sistema mono visión para obtener el campo visual del sistema.

Cámara inferior, ubicada en la parte inferior del UAV, empleada en definitiva para conseguir una mayor estabilidad del Dron, sin realizar maniobras de estabilizado en modo reposo del Dron, la calidad es muy inferior a la frontal y su color sigue siendo RGB.

Para el control de la altitud el sensor que incorpora, se trata de un sensor de ultrasonidos en la parte inferior del esqueleto del Dron, el cual emite dichos ultrasonidos hacia la superficie y mediante el cálculo automático de la latencia con la que se reciben estima a que altitud respecto de la superficie se encuentra en cada momento. (Fuentes Brea)

Ilustración 14: Ubicación sensor de altitud

El ARDrone dispone de un giroscopio de 2 ejes y un giroscopio eléctrico que se encargan de realizar las diferentes mediciones del giro, el alabeo y el cabeceo del dron.

Para el control de la orientación el dron emplea un sensor llamado magnetómetro, que junto a los tres giroscopios que posee se encargan del estabilizado del sistema.

4.1.1.1.5 GPS FLIGHT RECORDER

Debido a la importancia de tener la mayor precisión para realizar el control y la capacidad para conectar dispositivos mediante USB de nuestro Dron, Parrot pone a disposición un dispositivo compatible llamado GPS Flight Recorder, empleado para la obtención de coordenadas GPS como latitud y longitud, así como grabación de rutas de vuelo.

Este dispositivo de conecta por USB y mediante los protocolos de comunicación (se trabajará con ellos más adelante) y el sistema Wifi del Dron se comunicará con la base para el paso de información de este dispositivo. (ardronespain, 2016)

Tabla 1:Información técnica Parrot 2.0

Información Técnica	
Precisión GPS	78.74''
Capacidad	
Memoria Flash	4GB
Características Físicas	
Altura	3.1''
Anchura	1.5''

Para el desarrollo de nuestro sistema de control no se ha empleado este dispositivo debido a las limitaciones de la arquitectura de las librerías empleadas, pero si se ha empleado como objeto de estudio de diversas alternativas de control y estudio de nuevas herramientas para obtención de trayectorias punto a punto.

4.1.1.1.6 ACTUADORES

Como he mencionado anteriormente no existen sensores para calcular o estimar en qué posición o que movimiento se ha producido, pero si pueden obtenerse los valores exactos de sus ejes de rotación de tres dimensiones, así como la velocidad de desplazamiento basándose en un algoritmo de estimación a través de las imágenes capturadas por la cámara inferior. Finalmente disponemos de la información de su batería mediante un medidor de carga, que devuelve el valor en porcentaje.

Toda esta información es proporcionada por el AR Drone SDK a través de la estructura de librerías que incorpora el Dron llamada **LSIDrone** formado por un conjunto de librerías, de la que existen diferentes versiones con las que trabajar (explicado en profundidad más adelante).

Otros actuadores del ARDrone son las cuatro hélices con sus respectivos rotores instaladas sobre una cruceta central de metal, que varían su velocidad de rotación para realizar los diferentes tipos de movimientos. Estas hélices motoras pueden recibir una serie de instrucciones predefinidas como son el aterrizaje o el despegue con una

potencia máxima de 15 W y 28000 RPM en vuelo estático llegando a tener una velocidad de desplazamiento de 5 m/s. Estos cuatro motores se conectan al controlador electrónico que es el encargado de trasmitirle la potencia a través de una alimentación de 5V. (ardronespain, 2016)

4.1.2 NAVEGACIÓN DE ARDRONE 2

El AR Drone 2 se trata de un cuadricóptero y como los drones de su familia realiza los movimientos de forma muy similar, empleando los tres ángulos de navegación de Euler sobre los ejes X, Y, Z que explicamos con anterioridad (véase Funcionamiento del cuadricóptero).

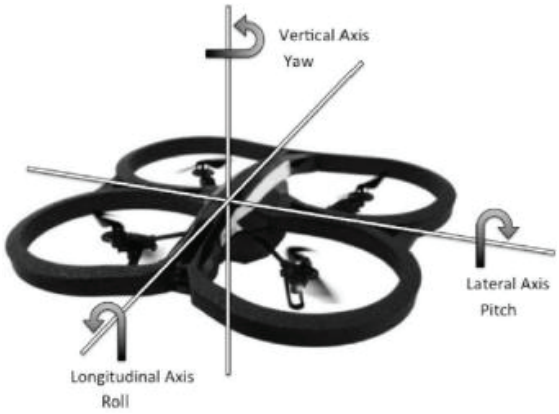


Ilustración 15: Ángulos de navegación

Pero a la hora de realizar un desplazamiento el sistema de velocidades tiene una serie de particularidades:

El avance y el retroceso lo realiza sobre el eje X, siendo el avance una velocidad negativa -Vx y el retroceso la inversa, una velocidad positiva Vx.

Para el desplazamiento lateral ocurre de forma similar, el desplazamiento hacia la derecha y la izquierda se realiza sobre el eje Y, y para el primer caso su velocidad Vy será positiva y para el segundo caso -Vy será negativa.

Para las rotaciones del eje sigue la política de movimiento del desplazamiento lateral, dYaw positivo rotación hacia la derecha y en caso contrario rotación negativa.

Para el aumento de altura H en un despegue o en caso contrario para un, siendo mayor si queremos ascender y menor si queremos descender.

4.2 ESTACIÓN BASE ARDRONE 2

El Ar Drone 2 tiene unas librerías que permiten el acceso a su firmware interno en diferentes sistemas operativos ya sea Windows como Linux para permitir la comunicación entre el UAV y la estación de operaciones, a través de la cual podremos realizar la infraestructura de control que establecerá la conexión con el dron a través del Wi-Fi que este genera, y la antena receptora Wi-Fi de la estación.

En este caso hemos utilizado un ordenador portátil HP-G62 con procesador i3 de segunda generación estas especificaciones del equipo de trabajo son más que suficientes para procesar la información que recibimos ya que no se trata de una comunicación masiva de información y al procesado para realizar la infraestructura de comunicación con el dron. El equipo está dotado de un sistema operativo Windows 10 y el entorno de desarrollo para robótica de los algoritmos que hemos desarrollado a través del programa Visual Studio 2013 que nos permitirá importar las librerías y realizar nuestro proyecto en un entorno de programación en C# ya que queremos disponer de un sistema de control propio, empleando como se mencionó anteriormente LSIDrone.

En esta versión de LSIDrone se permite acceder a la información que nos comunica el dron a través de la red Wi-Fi y poder trabajar con ella, además nos permite acceder a sus rotores y modificar sus movimientos, mandándole comandos de vuelo y control mediante el protocolo de comunicaciones de nuestro dron que se explicará más adelante.

5. PROTOCOLOS DE COMUNICACIÓN

El Ar Drone 2 establece una comunicación con cualquier dispositivo cliente que disponga de un controlador Wifi, de tal forma que en primer lugar el dron crea una red Wifi llamada ARDrone2 de acceso libre con una dirección IP. En segundo lugar, el cliente se conecta a la dirección de dicha red y solicita una dirección IP al servidor DHCP del dron.

El dron otorga una dirección IP al cliente que suele ser la IP del Dron sumándole uno o sumándole entre uno y cuatro números. Desde este momento el cliente podrá mandar peticiones a la IP del Dron y su puerto (Parrot, ARDrone Developer Guide).

5.1 PROTOCOLOS DE COMUNICACIÓN DEL DRON

El control del Dron se realiza a través de cuatro servicios de comunicación.

- Controlar y configurar el dron se realiza mandando comandos AT a través del protocolo de comunicaciones de la capa de transporte UDP a través del puerto 5556 con una latencia de trasmisión de los comandos en torno a 30 veces por segundo de forma regular. ¿En qué consisten estos comandos AT?, estos comandos son cadenas de texto que contienen las acciones enviadas al dron para el control de sus acciones, los cuales son creados de forma automática por las librerías de nuestro dron del proyecto LSIDrone y enviados al puerto de serie 5556 en paquetes UDP. Estos Strings se codifican en caracteres de 8-bit ASCII con un retorno de carro al final de cada mensaje para delimitar la información.

Cada mensaje se compone de el nombre o identificador, signo igual (=), una secuencia de números y de forma opcional una lista de argumentos separados por comas dependiendo del tipo de comando que se envía.

Un paquete UDP por contener uno o más comandos AT separados en distintas líneas, pero un comando AT no puede separarse en distintos paquetes UDP. (Parrot, ARDrone Developer Guide)

Un ejemplo de paquete UDP con dos comandos AT obtenido de la práctica sería:

AT*PCMD_MAG=21625,1,0,0,0,0,0,0<CR>AT*REF=21626,290717696<CR>

- Por otro lado, la información del dron como su posición, velocidad, rotación, estado… llamada NavData, se envía también mediante paquetes UDP al puerto 5554 por defecto del dron. Esta NavData incluye etiquetas para detectar la información y son enviados con una latencia de 15 veces por segundo aproximadamente.

- En tercer lugar, la información de video se envía mediante paquetes TCP al puerto 5555. Esta información es decodificada empleando el códec de las librerías del dron.

- Por último, el cuarto canal de comunicación se trata del llamado puerto de control que establece una comunicación mediante TCP al puerto 5559, para transmitir los datos críticos. Se utiliza principalmente para recuperar los datos de configuración y reconocer la información importante, como el envío y recepción de ACKs. (Parrot, ARDrone Developer Guide).

5.2 PROTOCOLO DE COMUNICACIONES MAVLINK

5.2.1 INTRODUCCIÓN

Anteriormente se ha explicado el modo de conexión entre las diferentes partes del sistema base y dron que emplearemos para el desarrollo del sistema de control del dron. Para realizar este sistema de control, y ejecutar la solución técnica objetivo es muy importante recabar una gran cantidad de información para conocer los diferentes valores de vuelo y el comportamiento del dron, por ello es de gran importancia obtener un mayor número de información para poder realizar un análisis de datos completo y realizar un sistema eficiente.

De esta forma tras muchas investigaciones, he descubierto la existencia de un protocolo de comunicación de vehículos aéreos no tripulados que está teniendo una gran aceptación y utilización durante los últimos años. Este protocolo es MavLink, y es integrable en un gran número de drones para realizar diferentes operaciones. En nuestro caso no es integrable a la plataforma LSIDrone que se explicará detalladamente en la siguiente sección, por ello se emplea en conjunto con un nuevo software OpenSource llamado QgroundControl que se explicará más adelante, en el que se integra este protocolo y nos permite obtener multitud de información de vuelo del dron (latitudes, altitudes, velocidades…) y diseñar trayectorias de vuelo autónomo punto a punto.

5.2.2 DESCRIPCIÓN DE MAVLINK

MavLink es un protocolo de comunicación para MAV (Micro Aerial Vehicles) que hoy en día se ha extendido a todo tipo de aviones no tripulados (tanto aéreos como terrestres).

El protocolo MavLink emplea una serie de librerías que empaquetan en mensajes unas cabeceras que están diseñadas para los UAV. Estas cabeceras emplean lenguaje C y son enviadas desde los puertos de serie del Dron a la estación base de forma bidireccional, en nuestro caso QgroundControl; dichos puertos del dron serán los mismos que en el protocolo de comunicaciones con el software LSIDrone, por ello no pueden funcionar de forma simultánea ambos sistemas. (Wikipedia, Wikipedia.org, 2016)

Las librerías que emplea dicho protocolo son muy ligeras y fáciles de integrar en cualquier sistema autónomo de vuelo o tierra como en cualquier estación base, generando una serie de mensajes que empaquetan tanto la estructura de datos como las cabeceras anteriormente mencionadas. Existen librerías de mensajes comunes pero una gran ventaja de este protocolo y sus librerías es la posible personalización de los mensajes. (qgroundcontrol.org, 2016)

El software MavLink tiene licencia LGPL (GNU Lesser General Public Licencense) lo que permite que sea utilizado tanto en proyectos closesource como opensource.

El coste computacional de MavLink es bajo, por lo que va a poder ser incluido en cualquier proyecto de forma no intrusiva sin tener que ser utilizada como parte principal del proyecto. Las librerías manejan tanto los parámetros, como las misiones y la telemetría de vuelo, mientras que el piloto automático solo se encarga de extraer los datos (Crespo, 2014).

A continuación, se muestra una tabla de rendimiento para nuestro Dron:

Tabla 2: Rendimiento del ARDrone 2.0

Velocidad de enlace	**Hardware**	**Tasa de refresco**	**Contenido**
42080 bps	AR Drone 2	20 Hz	38.400 – 57.600 bps
105200 bps	AR Drone 2	50 Hz	57.600 – 115.200 bps
210400 bps	AR Drone 2	100 Hz	128.000– 256.000 bps

Como se puede apreciar este protocolo fue totalmente orientado hacia dos propiedades: la velocidad y la seguridad de transmisión.

5.2.3 ESTRUCTURA DE PAQUETES MAVLINK

Cada paquete Mav está estructurado de la siguiente manera:

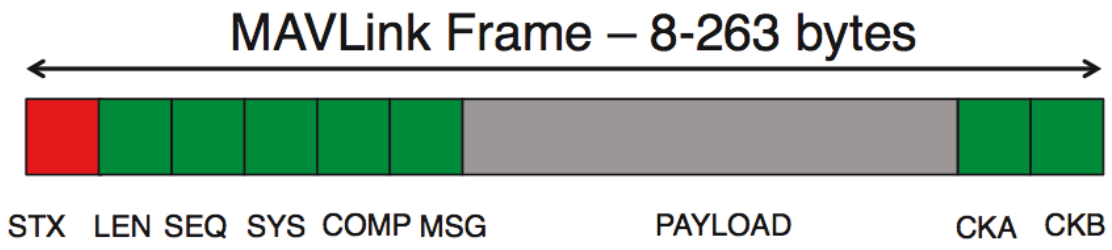


Ilustración 16: Formato paquete MavLink

A continuación, se muestra una tabla con la descripción de cada una de las partes del mensaje: (Gitbooks, 2016)

Tabla 3:Formato paquete MavLink

Partes del mensaje	Byte	Descripción
Cabecera	0	Indica que se inicia un mensaje nuevo
	1	Longitud del contenido
	2	Número de secuencia, permite detectar si se pierde un paquete
	3	ID del sistema de envío.
	4	ID del emisor
	5	ID del mensaje
Contenido	6 - (n + 6)	Contenido del mensaje
Checksum	(n+7) to (n+8)	ITU X.25/SAE AS-4 hash, excluido el inicio del mensaje, desde el byte 1...(n+6) Nota: El checksum también incluye MAVLINK_CRC_EXTRA (Número calculado a partir de campos del mensaje. Protege el paquete de ser decodificado de una versión diferente del mismo paquete, pero con diferentes variables)

La longitud mínima de un paquete Mav es de 8 bytes y la máxima de 263 bytes, como se muestra en la tabla comienza con un byte de inicio que indica el comienzo del mensaje, a continuación, se lee que longitud tiene el mensaje, y en caso de que se lea un nuevo inicio de mensaje y no se haya cumplido la longitud se considerará como una pérdida o modificación de alguno de los bytes del mensaje provocando que la comprobación del checksum sea fallida, en este caso se desprecia el mensaje y el receptor queda a la espera de un nuevo mensaje. Si el checksum coincide se procesa el paquete MavLink, y se vuelve a esperar a que se reciba un nuevo byte de inicio de mensaje. (Balasubramanian, 2016)

El cuerpo o contenido del mensaje será propio de la utilidad o datos que queramos extraer en nuestro caso del vuelo del AR Drone y será parte de la solución técnica de este proyecto.

5.2.4 SOFTWARE DE CONTROL DE VUELO MAVLINK, QGROUNDCONTROL

Como se ha comentado con anterioridad para el empleo del protocolo de comunicaciones MavLink es necesario emplear el software QgroundControl que permite una sencilla implementación de dicho protocolo, ya que vienen integradas sus librerías teniendo que realizar el diseño de la arquitectura de comunicaciones y la definición de mensajes para la obtención de datos y trayectorias de la solución técnica.

QGroundControl se trata de un software de control dron/tierra para el manejo de vehículos aéreos permitiendo visualizar y controlar un UAV durante su desarrollo y funcionamiento tanto de exteriores como de interiores. Su arquitectura es muy flexible y existen numerosas versiones compatibles con diferentes tipos de UAVs. Al tratarse de una aplicación de código abierto permite el desarrollo por parte de desarrolladores de todo el mundo pudiendo modificar su código y adaptarlo para diferentes controles de vuelo y utilidades, motivo principal del uso de esta plataforma para este proyecto.

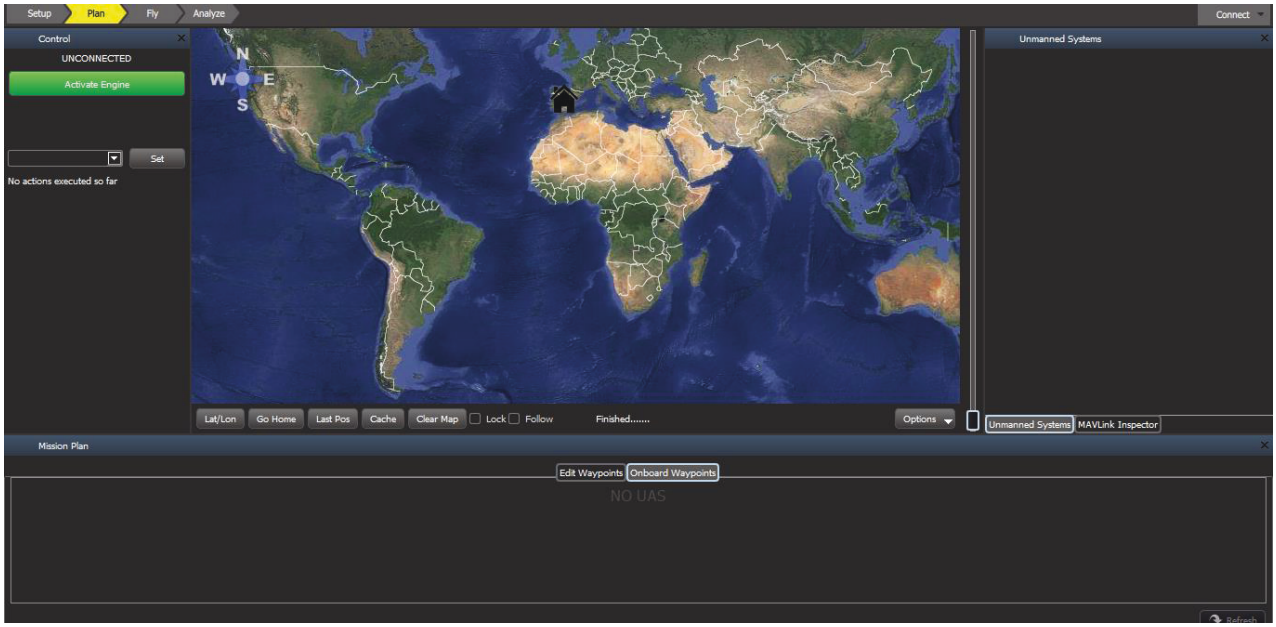


Ilustración 17: Interfaz QgroundControl-Plan

Sus principales características son:

- Plataforma OpenSource con integración de MavLink con capacidad de incluir microcontroladores ligeros.
- Emplea mapas 2D y 3D para la definición de planes de vuelo.
- Manipulación de puntos de referencia y parámetros de a bordo.
- Trazado a tiempo real de sensores y telemetría.
- Soporte para UDP y redes de malla.
- Compatible con distintos pilotos automáticos, entre ellos AR Drone 2.
- Su protocolo MavLink soporta hasta 255 vehículos simultáneos.

QgroundControl como se puede ver en sus características funciona a través de datos GPS para planificar el vuelo del Dron y obtener las coordenada y mapas de Google Hearth, por ello para la integración del sistema con nuestro Dron es necesario el empleo de una de las herramientas del desarrollador Parrot que ofrece para nuestro AR Drone 2, el GPS Flight Recorder. (QGroundControl, 2016)

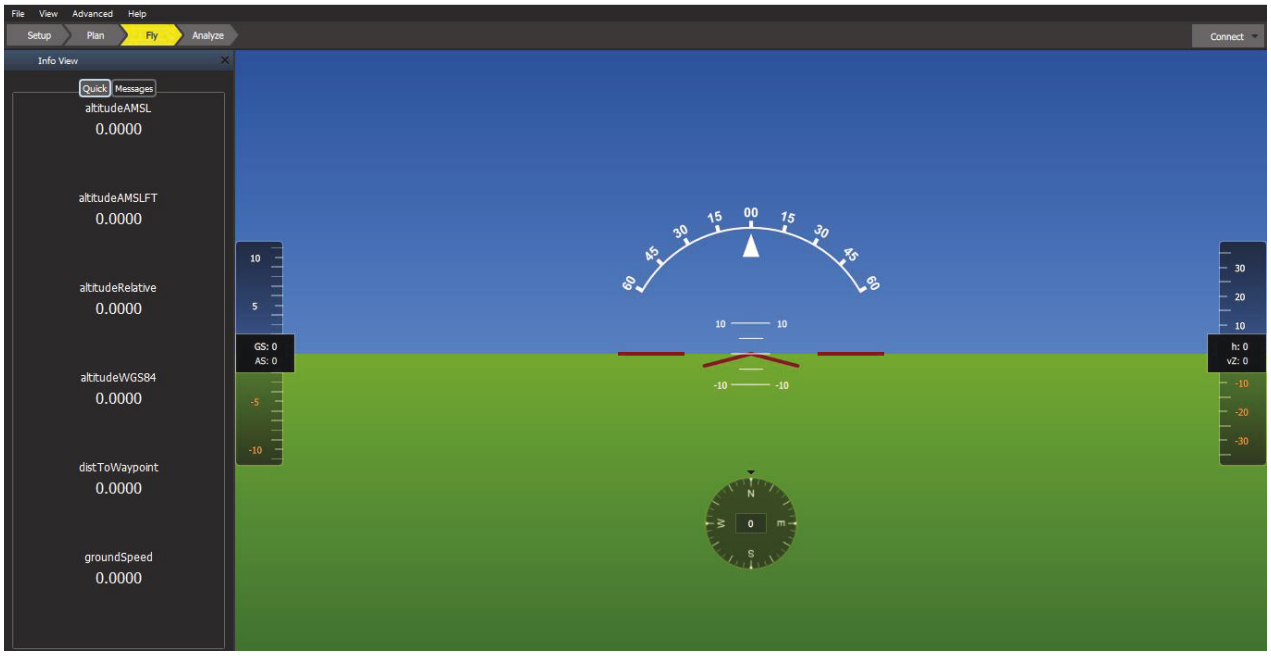


Ilustración 18: Interfaz QGroundControl-Fly

6. LSIDRONE

Este punto podría enmarcarse dentro de la tecnología empleada para el proyecto, pero debido a su gran importancia dentro del mismo y los desarrollos se enmarca como un nuevo punto.

El programa LSIDrone se trata de un software libre OpenSource desarrollado en la UC3M (por el grupo LSI, "Laboratorio de Sistemas Inteligentes") que internamente emplea las librerías y Apis suministradas por la compañía fabricante del ARDrone, Parrot, para permitir a los usuarios especializados en la robótica trabajar con el software del Dron dándonos esta potente interfaz de trabajo.

Este software se encuentra en diversas plataformas de programación como son Java, C y C++, pero nos hemos decidido por emplear la versión en C# que a pesar de no haber sido estudiado durante la carrera nos proporciona un rango más amplio y definido de datos que nos facilitará trabajar con la información emitida por el dron, además de permitir trabajar con facilidad con puertos, nos facilitará la comunicación con el dron. Además, al estar trabajando en Windows nos permitirá realizar una aplicación muy eficiente y rápida para Windows y poder trabajar con una interfaz gráfica. Es un lenguaje de programación similar a C++ con muchas de sus características, pero con la posibilidad de realizar una programación orientada objetos como en Java.

6.1 ESTRUCTURA DE LA PLATAFORMA LSIDRONE

Como se ha mencionado anteriormente se emplea el software LSIDrone debido a que contiene una serie de librerías y Apis para permitirnos acceder a las funcionalidades del dron y sus diferentes sensores para poder desarrollar nuestro software de control de vuelo autónomo.

Previamente hay que comprender las estructuras de las que disponemos a través de esta plataforma para poder realizar la comunicación entre el dron y la estación base, así como la obtención de la información de los sensores y las herramientas para ejecutar los comandos de vuelo. LSIDrone está formado por una serie de librerías que nos permiten obtener todas estas utilidades.

LSIDroneAviationInstruments: Contiene las librerías implementadas que nos permiten realizar la extracción de la información de los instrumentos del hardware del dron como l son sus sensores, así como el código que nos permite procesar dichos datos.

LSIDroneBasics: Incluye los ficheros de configuración del dron y su información legal.

LSIDroneCapture: Incluye las librerías para obtener las imágenes y videos capturados por el dron durante el vuelo.

LSIDroneControlLibrary: Contiene las librerías y archivos que permiten la conexión wifi con el dron, así como la extracción de los datos de vuelo obtenidos previo procesado y extracción de los instrumentos de aviación, para ser accesibles y trabajar con ellos. Además, esta librería nos permite tener acceso a los comandos de vuelo, como giros, elevaciones y descensos entre otros.

Y por último LSIDroneInterface que contendrá nuestro software de control de vuelo autónomo y la interfaz de vuelo cuya implementación parte como requerimiento necesario para la conexión entre el software y el dron, así como para la ejecución de nuestro software.

Además, para conseguir nuestro objetivo será necesario realizar un desarrollo de un sistema autónomo obteniendo en todo momento los parámetros e información de vuelo, siendo necesario un sistema de extracción, para finalmente dichos datos ser procesados y ejecutar el sistema de control, partiendo de un diseño detallado.

6.2 DISEÑO DE LA NUEVA INTERFAZ GRÁFICA PARA EL CONTROL DEL DRON

Originalmente Parrot nos aporta una interfaz gráfica para realizar vuelos de forma manual para el AR Drone 2. No obstante esta interfaz gráfica es muy sencilla y poco completa ya que tan solo nos permite realizar vuelos sencillos manejados de forma manual con joysticks que depende de la precisión del usuario. Otra de las alternativas de las que se dispone para realizar vuelos son las diversas aplicaciones Android para nuestro Parrot que son de código cerrado y nos limitan mucho a la hora de poder recabar información y poder realizar un control de vuelo autónomo y preciso.

Ilustración 19: Interfaz gráfica serie Parrot 2.0

Por ello para el entorno LSI Drone se ha diseñado una interfaz gráfica propia creando un nuevo proyecto asociado al LSIDrone llamado LSIDroneInterface, que nos permita recabar información visual en tiempo real de los parámetros de vuelo de nuestro Dron empleando las librerías que nos ofrece el software, además de una serie de comandos de control para realizar vuelos más complejos y autónomos.

En este proyecto realizaremos todo el desarrollo de nuestro software.

6.2.1 VARIABLES PREDEFINIDAS Y MODOS DE VUELO

Para el desarrollo de la interfaz es necesario comprender y conocer cuáles son las variables que se emplean para la realizar la funcionalidad de los nuevos botones de la interfaz y los modos asociado a otros de ellos.

En primer lugar, voy a explicar los diferentes modos de vuelos predefinidos en el software LSIDrone, tan solo se mencionará su funcionalidad puesto que más adelante en el sistema de control realizado se explicará cómo funcionan, ya que en este apartado queremos mostrar a que botones de la interfaz están asociados:

i) TakeOff, es el comando del sistema que envía al dron la funcionalidad de despegue del dron.

ii) Land, es el comando que se encarga de enviar la señal de aterrizaje al dron.

iii) Emergency, se encarga de realizar una parada de emergencia como su propio nombre indica en el caso de producirse una situación de riesgo o peligro produciéndose una detención del dron.

iv) FlatTrim, esta instrucción se encarga de realizar un reseteo de los valores de los sensores del dron en posición horizontal, es decir la inicialización de los sensores.

v) EnterHoverMode, mediante esta instrucción nuestro dron entrará en modo de suspensión en el aire, es decir se equilibrarán las velocidades de los rotores para mantenerse en equilibrio.

vi) LeaveHoverMode, se abandona el modo de equilibrio del dron y comenzará el movimiento.

Por otro lado, hay que mencionar las variables que se muestran con sus valores en nuestra interfaz:

1) dVx, esta variable privada del sistema es la encargada de determinar la velocidad en el eje X del Dron, es decir, determina cómo avanza o retrocede el dron. Es muy importante destacar que esta variable será negativa en el caso de que avance y positiva en el caso de retroceso.

2) dVy, esta variable se encarga de determinar la velocidad de desplazamiento lateral de nuestro dron, siendo positiva con desplazamiento a la derecha y negativa con desplazamiento a la izquierda.

3) dH, hace referencia a la altura de vuelo por defecto está definida a 1000, es decir un metro de altura, que será a la altura a la que realizaremos las pruebas de nuestro sistema de control.

4) dYaw, esta variable está medida en grados, e indica el número de grados de giro del dron.

5) Las variables Roll y Pitch, encargadas de medir el alabeo son el resultado de operaciones con la variación de altura y velocidades, que se explicarán más adelante.

6.2.2 BOTONES DE LA INTERFAZ GRÁFICA E IMPLEMENTACIÓN EN LSIDRONE

Dentro del software LSIDrone, en nuestro proyecto LSIDroneInterface se ha creado una clase denominada MainForm, donde se declararán todas las funciones encargadas del control del vuelo, la ejecución de los botones de vuelo y obtención de información, por ello dentro de esta clase hemos definiciones las siguientes funciones para cada botón implementado.

6.2.2.1 BOTONES DE ATERRIZAJE Y DESPEGUE

Para esta funcionalidad se ha implementado la función btnControlTakeLand_Click (), que será empleada de manera indistinta para el aterrizaje y el despegue.

Esta función comprueba si está volando el dron, con una de sus variables globales del sistema y en el caso de que no sea así al ejecutar el botón, establece la altura dH a 2000, 2 metros, y las velocidades del eje x y del eje y a 0, para equilibrar el dron, entonces en este momento entrará en hover, ejecutando la función del sistema EnterHoverMode(). En caso contrario ejecutará el comando Land ().

Ilustración 20: Botón aterrizaje despegue

6.2.2.2 BOTONES DE AVANCE O RETROCESO

Para la ejecución de dichos botones se han definido dos funciones, en primer lugar, el botón de avance ejecutará la función btnForward_Click () que en primer lugar sale del modo hover LeaveHoverMode () establecerá la altura a la altura por defecto de 1 metro y avanzará a una velocidad constante de -500, dVx=-500.0, expresado en milímetros segundos, es decir 5 metros/segundo.

Ilustración 21: Botón avance

Y en el caso del retroceso la función que ejecuta el botón es btnBack_Click () cuyo funcionamiento es similar al de avance, pero cambiando el signo de dVx a positivo, es decir, dVx=-500.0.

Ilustración 22: Botón retroceso

6.2.2.3 BOTONES DE DESPLAZAMIENTO LATERAL

Para realizar desplazamiento hacia la derecha se ha creado la función btnYRight_Click () que ejecuta una salida del modo de equilibrio del dron LeaveHoverMode (), mantiene la altura a 1 metro y realiza un aumento de la velocidad de eje y dVy a 500 milímetros por segundo, dVy=500.0.

Ilustración 23: Botón desplazamiento lateral

En el caso de un giro a la izquierda el proceso es el mismo, pero con diferente signo en la velocidad del eje Y.

6.2.2.4 BOTONES DE ROTACIÓN

Para la rotación dYaw hemos definido giros de 90 grados hacia la izquierda o hacia la derecha siguiendo la política de que un giro hacia la derecha será dYaw= dYaw + 90.0 y hacia la izquierda dYaw = dYaw - 90.0, de esta forma controlamos los giros previos que se han realizado siendo dYaw los grados de la orientación hacia los que se está moviendo el Dron, siendo inicialmente una orientación a 0 grados. Como en las anteriores funciones debemos salir previamente del modo hover LeaveHoverMode ()

Ilustración 24: Botones de rotación

6.2.2.5 BOTONES DE VARIACIÓN DE ALTURA

Para el funcionamiento de los botones de aumento de altura y disminución hemos creado las funciones btnUp_Click () y btnDown_Click () que se encargan de aumentar o disminuir dH de forma constante una distancia en el eje Z de medio metro dH=dH+500.0

Ilustración 25: Botones de variación de altura

6.2.2.6 BOTONES DE VUELO AUTÓNOMO

Los anteriores botones han sido implementados para realizar vuelos manuales, y realizar pruebas, pero el objetivo de este proyecto es realizar un sistema autónomo de control gran precisión por ello esta es la función y botón a ejecutar más importante.

Las funciones de estos botones serán explicadas más adelante ya que serán las encargadas de ejecutar el vuelo autónomo y el sistema de control y corrección de nuestro Dron, en particular nos centraremos en la función de control que ejecuta el botón que se muestra a continuación ya que nuestro sistema de control realizará un vuelo autónomo sobre los ejes X e Y realizando modificaciones del Yaw. La función que llamará al algoritmo de vuelo autónomo será btnAutonomousFlight ()

Ilustración 26: Botón de vuelo autónomo

6.2.2.7 PARÁMETROS DE VUELO

En la interfaz se muestran en tiempo real los valores de las velocidades, ángulos de giro, altura, valores de los sensores, entre otros valores necesarios en el vuelo autónomo los cuáles serán descritos más adelante, todos estos valores se actualizan de forma continua mediante la definición de un Timer de ejecución llamado tmrStatusUpdate:

```
tmrStatusUpdate = new DispatcherTimer();
tmrStatusUpdate.Interval = new TimeSpan(0, 0, 0, 0, 150);//50
tmrStatusUpdate.Tick += new EventHandler(tmrStatusUpdate_Tick);
```

Ilustración 27: Definición de un timer

Que se activa cada 50 milisegundos y obtiene estos valores de los valores de las variables del sistema que se van modificando al realizar los diferentes comandos de vuelo descritos anteriormente llamando al manejador tmrStatusUpdate_tick () que activa la función update () que se encarga de pintar en la interfaz dichos valores.

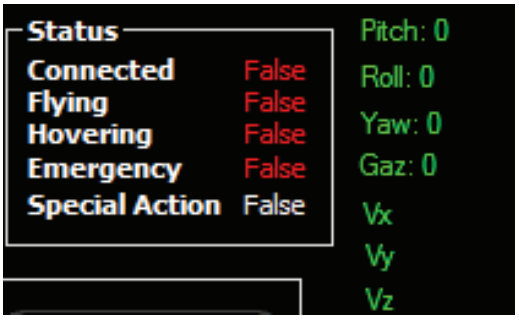


Ilustración 28: Cuadro de datos de vuelo

Además de todas estas variables y botones para el control del vuelo de la interfaz gráfica, el software LSIDrone contiene una serie de librerías y botones predefinidos para el control de la batería, encendido y apagado del Dron, emergencia del Dron y una serie de gadgets que obtienen los valores de los sensores del Dron para conocer su alabeo y nivel de estabilizado del Dron.

6.2.3 DISEÑO DE LOS BOTONES Y ELEMENTOS DE LA INTERFAZ

Tras haber definido las funciones de ejecución y los diferentes botones que compondrán nuestra interfaz gráfica, hay que explicar cómo se crean dichos botones e interfaz.

Para ello dentro de nuestro proyecto una de las grandes posibilidades que nos da c# es la posibilidad de generar elementos gráficos, por lo que nuestra clase MainForm.cs se divide en la parte de código e implementación y la parte gráfica de la clase, por lo tanto, nuestra clase MainForm.cs se divide en dos clases parciales MainForm.cs y MainForm.Designer.cs en la que se define el diseño de nuestra interfaz y los botones.

Dentro de esta "Clase Parcial" existe una función principal llamada InitializeComponent (), que contiene la definición de los diferentes elementos de la interfaz, declarando en primer lugar las divisiones de la interfaz, los botones que vayamos a emplear y a continuación sus nombres y diseño.

- Para el caso del botón de vuelo autónomo:

1. Definición del botón:

```
this.btnAutonomousFlight = new System.Windows.Forms.Button();
```

Ilustración 29: Implementación de un nuevo botón

2. Diseño del botón

```
this.btnAutonomousFlight.BackgroundImage =
    ((System.Drawing.Image)(resources.GetObject("btnAutonomousFlight.BackgroundImage")));
this.btnAutonomousFlight.BackgroundImageLayout =
    System.Windows.Forms.ImageLayout.Zoom;
this.btnAutonomousFlight.FlatAppearance.MouseDownBackColor =
    System.Drawing.Color.Black;
this.btnAutonomousFlight.FlatAppearance.MouseOverBackColor =
    System.Drawing.Color.Silver;
this.btnAutonomousFlight.FlatStyle =
    System.Windows.Forms.FlatStyle.Popup;
this.btnAutonomousFlight.ForeColor =
    System.Drawing.Color.Black;
this.btnAutonomousFlight.Location =
    new System.Drawing.Point(475, 30);
this.btnAutonomousFlight.Name = "btnAutonomousFlight";
this.btnAutonomousFlight.Size =
    new System.Drawing.Size(100, 70);
this.btnAutonomousFlight.TabIndex =
    148;
this.btnAutonomousFlight.UseVisualStyleBackColor = false;
```

Ilustración 30: Implementación del diseño de un botón

3. Y finalmente debemos asignar la función correspondiente a la ejecución del botón como se ve en la última línea de la figura anterior.

```
this.btnAutonomousFlight.Click += new System.EventHandler(this.btnAutonomousFlight_Click);
```

Ilustración 31: Asignación de una función a un botón

6.2.4 IMAGEN FINAL DE LA INTERFAZ

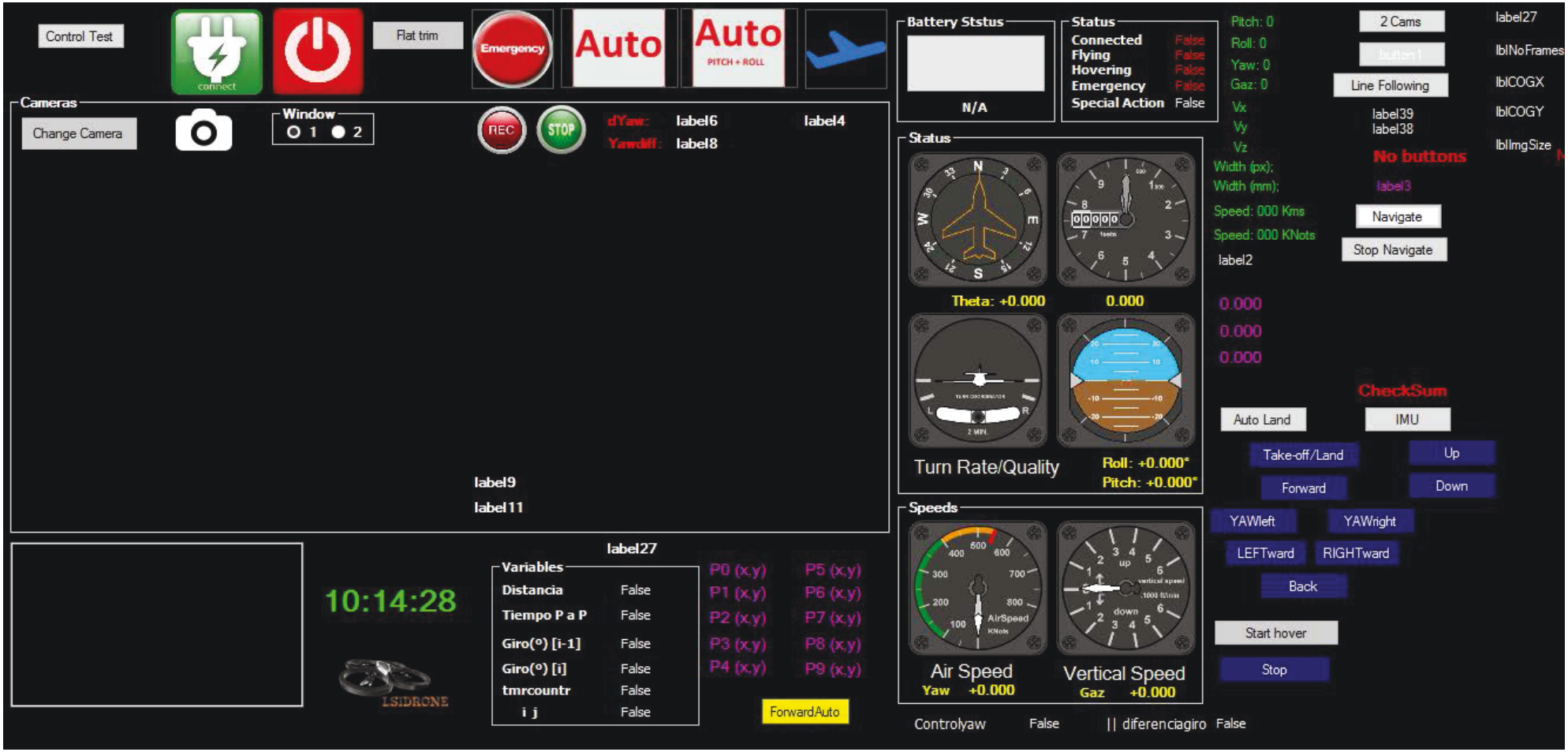


Ilustración 32: Interfaz gráfica final

6.2.5 DIAGRAMA COMUNICACIÓN LSIDRONE Y ARDRONE 2

Por lo tanto, tras haber especificado la comunicación entre el dron y nuestra estación base la cual contiene el software LSIDrone (ver Protocolos de comunicación) y nuestra nueva interfaz de control del dron, podemos representar el esquema de comunicaciones entre las distintas partes de nuestro sistema, sobre el que comenzaremos el diseño e implementación de la solución técnica.

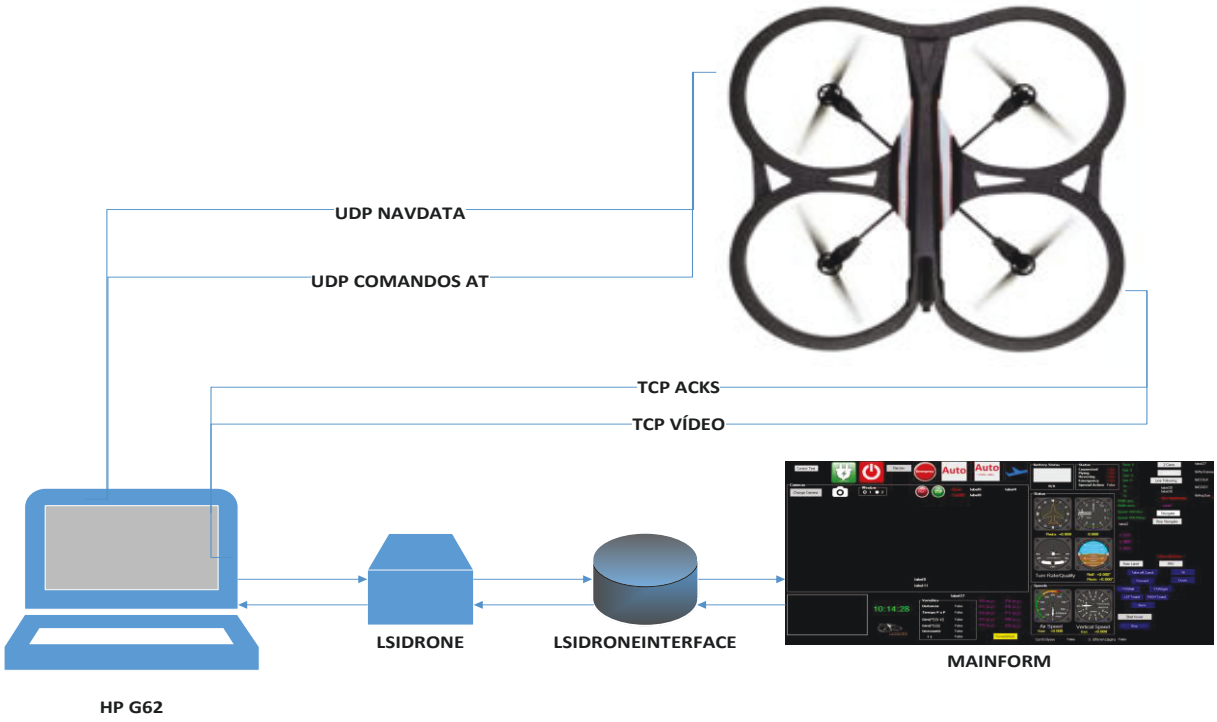


Diagrama 3: Comunicación de protocolos dron LSIDrone

7. DESARROLLO DE LA SOLUCIÓN TÉCNICA

Llegado a este punto nos encontramos en el instante de partida para el desarrollo de nuestro Sistema de control, en este momento disponemos del software LSIDrone con el Proyecto LSIDroneInterface en el cual se encuentra la clase MainForm sobre la que implementaremos nuestro sistema de control que ya dispone de las funciones mencionadas con anterioridad para el desarrollo de la interfaz gráfica de control de nuestro software y accesibilidad para todas las librerías del software. Además, ya conocemos las diferentes tecnologías que empleamos en nuestro software de control y como se comunican las diferentes partes que compondrán nuestra solución técnica.

Por ello a continuación se va a detallar el desarrollo de la fase de diseño del sistema de control autónomo que será implementada posteriormente para cumplir ya que para un buen desarrollo de software es imprescindible realizar un diseño previo.

7.1 DISEÑO

En primer lugar, deberemos definir el tipo de control de vuelo autónomo que querremos emplear en nuestro sistema de control, en este caso se tratará de un sistema de control de lazo abierto.

7.1.1 SISTEMA DE CONTROL EN LAZO ABIERTO

En estos sistemas la señal de salida no influye sobre su regulación. Se obtienen los datos de entrada y se ejecuta el proceso de control.

Es aquel sistema en que solo actúa el proceso sobre la señal de entrada y da como resultado una señal de salida independiente a la señal de entrada, pero basada en la primera. Esto significa que no hay retroalimentación hacia el controlador para que éste pueda ajustar la acción de control. Es decir, la señal de salida no se convierte en señal de entrada para el controlador.

Por ejemplo, un sistema de control de riego en lazo abierto tiene un temporizador que lo pone en marcha todos los días a una determinada hora, es decir estos sistemas requieren de un sistema de temporizadores o activadores de señal, para su activación.

Para nuestro caso la entrada estará definida por una serie de coordenadas 'X' e 'Y', junto con el número de puntos que define la trayectoria, y los valores de control de vuelo, su velocidad, ángulos de giro, altura…. Una vez el sistema recibe dichos parámetros los procesa de tal manera que sea capaz de llegar al siguiente punto objetivo y modifica los valores de control para poder completar su movimiento. Una vez llegue a dicho punto vuelve a recalcular dichos valores de control para conseguir llegar al siguiente punto.

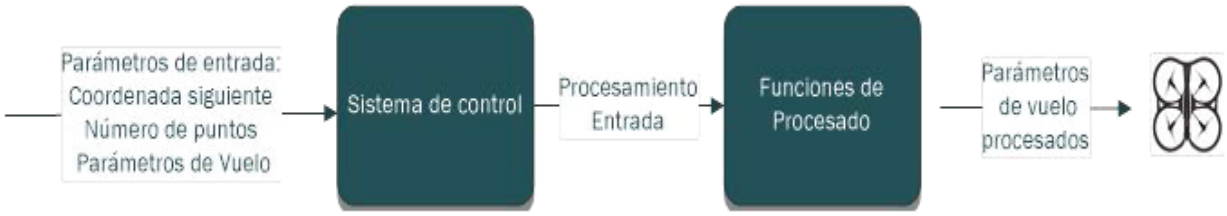


Diagrama 4: Sistema de control Lazo abierto

7.1.2 TEMPORIZADORES

Al tratarse de un sistema de control en lazo abierto será necesario establecer un sistema de temporizadores o activadores de las distintas funcionalidades del sistema de control, con una prioridad determinada dependiendo de cómo queramos que funciones nuestro sistema.

En este caso sería necesario emplear 5 temporizadores:

- Temporizador para la interfaz gráfica que muestre los valores de vuelo en tiempo real. Será interesante inicializarlo junto al sistema y que esté en continuo funcionamiento, es decir que se active cada muy poco tiempo para obtener en la interfaz gráfica diseñada los valores de vuelo en cada momento.

- Temporizador para el reloj del sistema, para saber el tiempo empleado en la trayectoria de vuelo. Activará el reloj para calcular el tiempo.

- Temporizador para el sistema autónomo. Este temporizador se deberá activar al pulsar sobre el botón de "Auto" de nuestra interfaz y se ejecutará hasta el fin de la trayectoria.

- Temporizador para la obtención de los datos de vuelo, es decir este activador se ejecutará cada poco tiempo para conocer en el punto del eje de coordenadas en que se encuentra en cada instante.

- Temporizador del sistema de control, será el encargado de activar el sistema que realizará las correcciones de vuelo dependiendo de la entrada de datos en cada instante.

7.1.3 SISTEMA DE VUELO AUTÓNOMO

Una vez ya hemos determinado los activadores de las diferentes partes de nuestro sistema, será preciso definir qué sistema de vuelo autónomo queremos definir para realizar nuestro sistema de control. Desde la parte de diseño se ha decidido que nuestro sistema de vuelo sea emplear un algoritmo por el cual nuestro dron seguirá una trayectoria definida en el eje X y en el eje Y siguiendo un ángulo de navegación Yaw, es decir el movimiento de guiñada produciéndose rotaciones sobre el eje vertical imaginario del dron sobre el plano X e Y (Plano horizontal).

El sistema de coordenadas para la definición de trayectorias se tratará de un Sistema de Referencia Local basado en el modelo WGS84 (Ver Sistemas de posicionamiento actuales) partiendo de la posición (0,0) con el dron en modo Land (en el suelo), con una orientación inicial 0 debido a las características del dron, siendo esta orientación 0 la orientación de la parte delantera del dron que está en dirección al eje X positivo. De esta manera el dron partirá de la posición inicial y la orientación inicial siendo indiferente si el dron apunta hacia el norte, sur, este u oeste cada vez que reciba una nueva trayectoria.

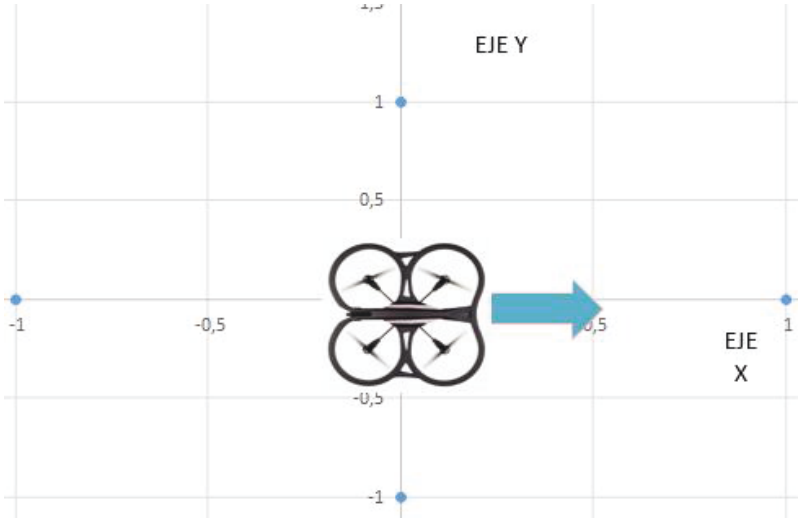


Ilustración 33:Sistema de referencia local

Una vez tenemos definido el vuelo autónomo sobre el que vamos a trabajar deberemos realizar un sistema que sea capaz de leer los parámetros de entrada, es decir los valores de los ejes, las coordenadas que debe de seguir y sus velocidades y una vez realizada la lectura de los valores de entrada determinar los movimientos que realizará para conseguir llegar al siguiente punto.

7.1.3.1 ESTRUCTURA DEL ALGORITMO DE VUELO AUTÓNOMO

En primer lugar, deberemos leer los puntos de la trayectoria de vuelo, para ello sería necesario crear un algoritmo de lectura de puntos, de la forma que consideremos oportuna, para nuestro diseño se realizará una lectura desde un fichero en formato de texto externo al software que estará formado por tres líneas, la primera de ellas hará referencia al número de puntos de la trayectoria, la segunda de ellas a las coordenadas X de cada punto, y por último las coordenadas Y de los puntos.

Una vez realizado esto será necesario almacenar dichos puntos para ser utilizados durante todo el sistema.

A continuación, deberemos obtener los valores Pitch, Roll y Yaw, los cuales deberán ser tratados por nuestro sistema y posteriormente procesados para que sean ejecutados por el Dron. Una vez llegado a este punto se deberá realizar el conjunto de posibles movimientos del Dron dependiendo de la situación del siguiente punto de coordenada.

Llegados a este punto se deberá realizar la lectura de cada punto, en primera instancia del primero de los puntos almacenados, teniendo en cuenta diferentes parámetros para decidir qué acción realizar, por ello vamos a diferenciar cuatro distintas fases de movimiento del dron para cada movimiento.

En primer lugar, la **fase de cálculo de parámetros** de vuelo, en esta fase se realizarán una serie de operaciones matemáticas que nos servirán para determinar qué acción elegir, las principales serán el tiempo que se estime tardará el dron en llegar al punto destino, la distancia a dicho punto y el ángulo beta que hace referencia al ángulo que existe entre el punto inicial y el punto final respecto del eje Y.

Una vez se han calculado dichos parámetros comenzará la segunda fase, a la que llamaremos **fase de orientación de vuelo,** en este momento deberemos salir del modo reposo del dron, es decir salir del modo hover, definiremos las velocidades de avance o retroceso del Dron para realizar el movimiento, y el giro Yaw que deberá hacer para

orientarse hacia el punto destino, este giro como ya sabemos se realiza sobre el eje Y orientando el dron en el eje X. En este momento deberemos empezar a recabar los datos de vuelo del Dron, es decir llamar al activador de la extracción de datos e ir obteniendo la trayectoria que se va realizando en cada instante, desde este momento estaremos obteniendo datos constantemente.

Una vez finalizada la fase de orientación de vuelo, volveremos a entrar durante un breve periodo de tiempo en modo reposo, hover, para a continuación realizar el movimiento hacia el siguiente punto.

Este movimiento se realizará en la fase cuatro del movimiento o **fase de avance** que consistirá en una vez haber orientado nuestro dron, saldremos del modo hover y comenzaremos a avanzar hacia delante en todo momento a una velocidad que determinaremos teniendo en cuenta el tiempo que se ha estimado hasta llegar al punto y la distancia a la que se encuentra.

El movimiento será siempre de avance ya que al tener en cuenta la ubicación del punto, el dron irá realizando rotaciones para siempre estar orientado hacia el punto y no tener que realizar de esta forma movimientos de retroceso.

Y, por último, una vez se haya estimado que se ha llegado al destino, volveremos al modo de reposo del dron de la primera fase y procederemos a leer el siguiente punto de la trayectoria en caso de haber más.

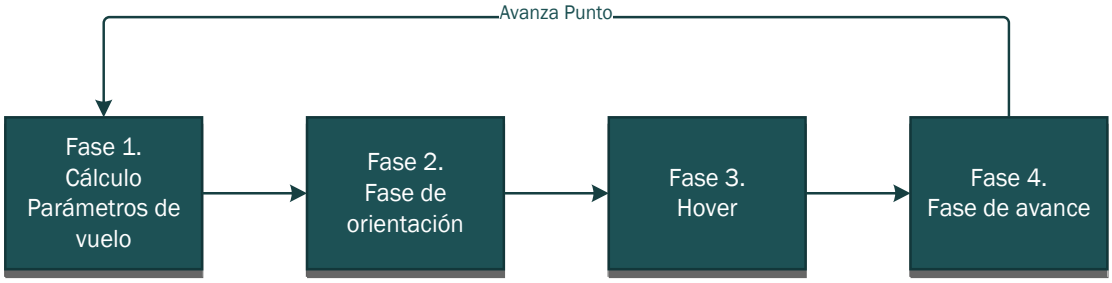


Ilustración 34: Fases del sistema

Este sistema de vuelo será igual para cada uno de los casos de vuelo que se van a analizar a continuación modificando los parámetros de vuelo y el giro que se realizará dependiendo del caso en el que estemos. Los casos de vuelo que se han detectado son los siguientes:

Tabla 4: Casos de vuelo autónomo

Caso	Posiciones X[i] e Y [i] respecto de X[i-1] e Y [i-1]
Caso 1	X[i] < X[i-1] -> Y[i] > Y[i-1]
Caso 2	X[i] > X[i-1] -> Y[i] > Y[i-1]
Caso 3	X[i] < X[i-1] -> Y[i] < Y[i-1]
Caso 4	X[i] > X[i-1] -> Y[i] < Y[i-1]
Caso 5	X[i] = X[i-1] -> Y[i] = Y[i-1]
Caso 6	X[i] > X[i-1] -> Y[i] = Y[i-1]
Caso 7	X[i] < X[i-1] -> Y[i] < Y[i-1]
Caso 8	X[i] = X[i-1] -> Y[i] < Y[i-1]
Caso 9	X[i] = X[i-1] -> Y[i] > Y[i-1]

7.1.4 SISTEMA DE OBTENCIÓN DE DATOS

Anteriormente hemos mencionado que se van a extraer una serie de datos de forma continua durante las diferentes fases del sistema de vuelo autónomo para conocer en cada momento los diferentes parámetros de vuelo como son sus velocidades, giros que va realizando, orientación actual del dron y coordenadas reales del dron en cada momento. La finalidad de este sistema que vamos a diseñar será recabar la suficiente información en cada momento para calcular las diferentes tasas de error respecto a las trayectorias que debiese llevar nuestro dron para llegar al punto destino exacto y las coordenadas por las que verdaderamente está pasando ya sea por diferentes factores externos o el acarreo de errores en la estimación de la trayectoria, tiempo, velocidad y giro durante las diferentes fases.

La obtención de estos datos se realizará de forma matemática ya que los diferentes sensores del dron y sus herramientas no nos proporciona la posición de referencia local en cada instante y su herramienta GPS Flight Recorder tan solo nos permite obtener los valores de su posicionamiento GPS global, pero no podemos trabajar con ellos para realizar un sistema de control eficiente y local.

Este sistema de extracción de datos se realizará mediante la activación de un temporizador en el momento que comienza al finalizar la fase de orientación del dron. En este momento deberemos comenzar a obtener los valores necesarios para el sistema de control.

En primer lugar, se tendrá en cuenta si el dron está en la fase de avance o en la fase de orientación ya que en la fase de colocación tan solo iremos actualizando los valores de las velocidades del dron y su giro Yaw, pero no se actualizarán las coordenadas puesto que no se realiza un movimiento de avance.

En el momento que comenzamos a avanzar empezaremos a actualizar los diferentes valores que emplearemos en nuestro sistema de control del dron, estando por un lado los valores ideales, aquellos valores que deberían tener los parámetros de vuelo, así como las coordenadas antes de llegar al punto destino y los valores reales que son aquellos valores producto del sistema de vuelo del dron.

En la tabla siguiente se muestran las variables que se extraen y emplearan en el sistema de control.

Tabla 5: Variables de vuelo

VARIABLES REALES E IDEALES
Velocidad Vy
Velocidad Vx
Coordenada X
Coordenada Y
Ángulo de giro

Estas variables se almacenarán y serán utilizadas por nuestro sistema de control como variables de entrada cada vez que se active la función de obtención de datos, de tal manera que al obtener los datos de vuelos del instante i, se ejecutará el sistema de control.

7.1.5 SISTEMA DE CONTROL

Tras haber realizado la fase de extracción de datos, de activará el temporizador de la función de control, está función recibirá las variables tanto reales como ideales, y las procesará.

Este procesado consiste en tras haber generado un sistema de control inteligente basado en reglas por el que se contemplarán una serie de casos prácticos, generando conjunto de reglas que tengan como entrada tanto los valores reales como ideales, para los que dependiendo de los valores de la entrada se ejecutará un conjunto de reglas determinado produciendo una salida que estará formada por la modificación o variación de las variables recibidas como entrada. Esta salida será comunicada al dron para realizar las modificaciones y correcciones en la trayectoria finalmente. En la fase de implementación entraremos con más detalle en este sistema de reglas de control.

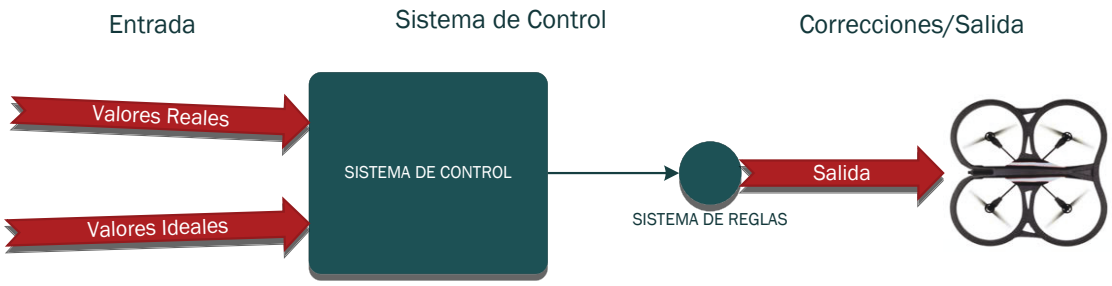


Diagrama 5: Sistema de control

7.1.6 DIAGRAMA DEL DISEÑO DEL SISTEMA DE CONTROL DE VUELO AUTÓNOMO

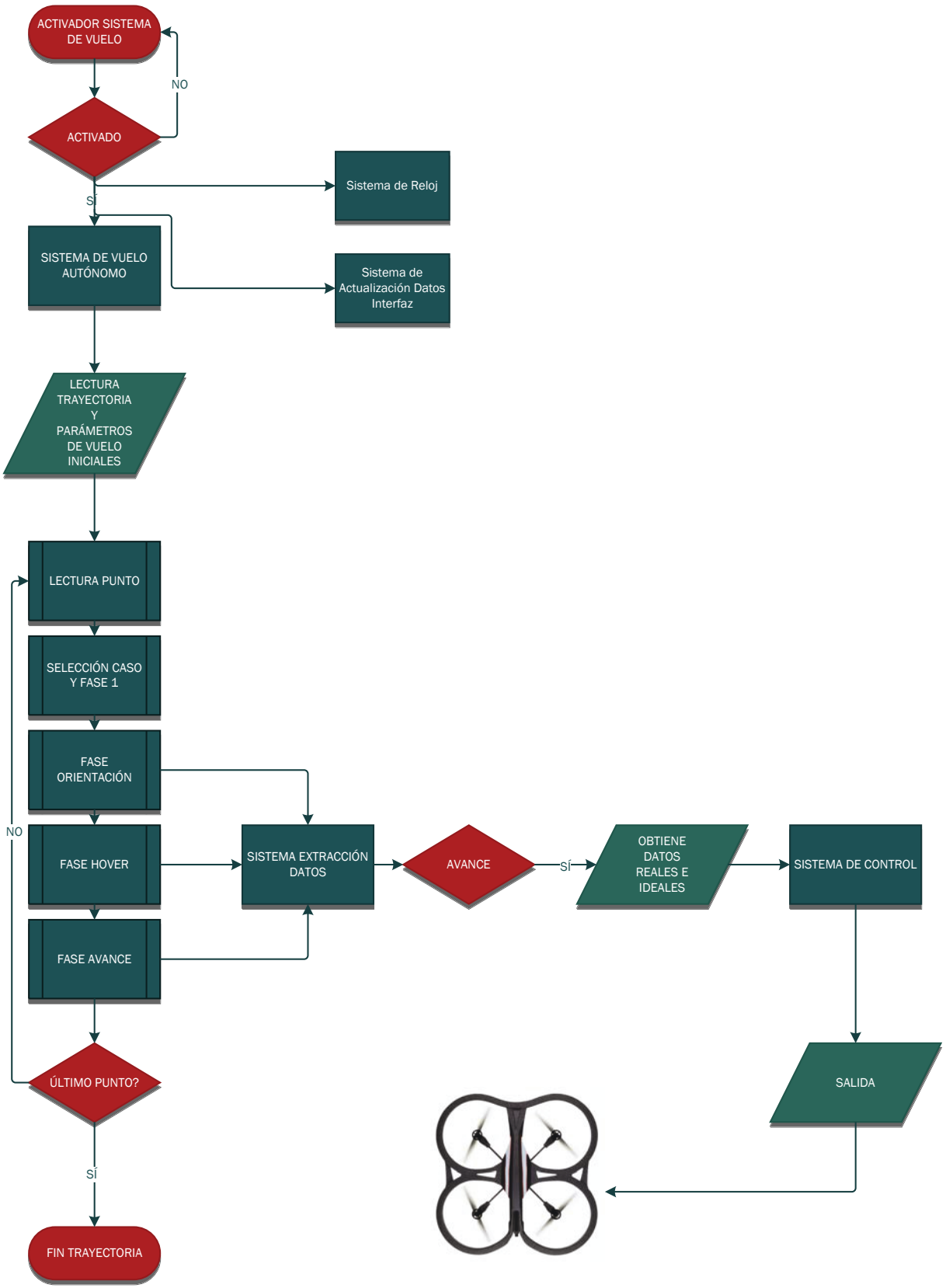


Diagrama 6: Diagrama del diseño del sistema de control de vuelo autónomo

7.2 IMPLEMENTACIÓN

Una vez hemos realizado el diseño y definido las funcionalidades de nuestro Sistema, se ha llevado a cabo la implantación en el software LSIDrone en concreto en la clase MainForm.cs de las diferentes partes que engloban dicho sistema. El lenguaje empleado como se mencionó anteriormente es C#.

7.2.1 TEMPORIZADORES

Por ello en primer lugar se han definido los diferentes activadores o temporizadores que activan las diferentes funcionalidades de nuestro sistema. En la fase de diseño se definieron cinco diferentes. Para ello se ha empleado la estructura de señales de C# llamada Dispatcher para la cual se define el tiempo automático de activación y el evento que dispara.

Dispatcher que actualiza los valores de vuelo en la interfaz gráfica:

```
tmrStatusUpdate = new DispatcherTimer();
tmrStatusUpdate.Interval = new TimeSpan(0, 0, 0, 0, 150);
tmrStatusUpdate.Tick += new EventHandler(tmrStatusUpdate_Tick);
```

Dispatcher que activa la señal de reloj.

```
tmrClock = new DispatcherTimer();
tmrClock.Interval = new TimeSpan(0, 0, 0, 0, 150);/
tmrClock.Tick += new EventHandler(tmrClock_Tick);
```

Dispatcher que activa el modo de vuelo autónomo.

```
tmrTestControl2 = new DispatcherTimer();
tmrTestControl2.Interval = new TimeSpan(0, 0, 0, 0, 500);//500
tmrTestControl2.Tick += new EventHandler(tmrTestControl2_Tick);
```

Dispatcher que activa la función de activación de datos.

```
tmrControl2 = new DispatcherTimer();
tmrControl2.Interval = new TimeSpan(0, 0, 0, 0, 200);
tmrControl2.Tick += new EventHandler(tmrObtainActualValues_Tick);
```

Dispatcher que activa la función del sistema de control.

```
tmrControl3 = new DispatcherTimer();
tmrControl3.Interval = new TimeSpan(0, 0, 0, 0, 30);
tmrControl3.Tick += new EventHandler(ControlSystem);
```

Esta estructura de datos como se observa declara el número de ticks o intervalo de tiempo en el que volverá a activarse la función del temporizador tras la primera llamada al temporizador y la función o método que se activa al ejecutarse.

Una vez implementada la estructura de señales para la ejecución de nuestro sistema se han implementado las diferentes funciones que lo forman.

7.2.2 FUNCIÓN DE ACTIVACIÓN DE RELOJ

Para esta función se ha definido el método tmrClock_Tick (), que implementa en el campo del reloj de sistema de nuestra interfaz gráfica definido previamente en MainForm.Designer.cs como lblClock el tiempo actual en un formato hora/minuto/segundo desde que se activa la señal.

```
private void tmrClock_Tick(object sender, EventArgs e)
{
    lblClock.Text = DateTime.Now.ToString("hh:mm:ss tt");
}
```

Esta función se ejecutará constantemente hasta el final de la trayectoria.

7.2.3 ACTUALIZACIÓN DE LOS VALORES DE LA INTERFAZ GRÁFICA

Se ha definido una función denominada tmrStatusUpdate_Tick () que se activará al iniciar el sistema, esta función llama a un método llamado UpdateStatus () que se encarga de actualizar los valores de vuelo actuales de nuestro dron en la interfaz gráfica.

Para esta función y a partir de este momento se utilizarán una serie de variables globales que se encuentran dentro de la clase del proyecto LSIDrone clsARDroneGeneralParams que hemos definido la cual contiene aquellos valores que tiene el cuadricóptero en ese momento en el IMU (unidad de medida inercial) es decir los valores resultado de los sensores y rotores del dron.

Para actualizar estos valores hay que acceder a la librería del dron dentro del proyecto que se llama NavigationData, una vez tenemos el objeto de esta clase podemos ir definiendo todos los valores de vuelo en clsARDroneGeneralParams que emplearemos en la función UpdateStatus para actualizar los valores de vuelo de la interfaz gráfica y poder usar estos parámetros en cualquier parte del sistema de forma actualizada ya que al ser un temporizador se ejecutará constantemente y tendremos acceso a los parámetros de vuelo actualizados en todo momento.

Una vez obtenidos estos parámetros generales tendremos que escribirlos en los campos de cada valor de la interfaz gráfica para poder ver de forma visual continuamente los valores de vuelo. La implementación realizada en primer lugar comprueba si el dron está conectado, si está desconectado resetea los valores de vuelo del Dron a cero.

```
private void UpdateStatus()
{
    if (!droneControl.IsConnected)
    {
        lblCamera.Text = "No picture";
        pgbarBatteryStatus.Value = 0;
        labelStatusBattery.Text = "N/A";

        lblRoll.Text = "Roll: +0.000°";
        lblPitch.Text = "Pitch: +0.000°";
        lblAltitude.Text = "0.000";
        lblSpeedKm.Text = "Speed: 000 Kms";
        lblSpeedKnts.Text = "Speed: 000 KNots";
        lblTheta.Text = "Theta: +0.000";

        imgbxColorCam.Visible = false;
        imgbxDetectionCam.Visible = false;
    }
```

En caso contrario obtiene los valores de la librería mencionada anteriormente y los almacena en clsARDroneGeneralParams y los escribe en los campos de la interfaz gráfica.

```
else
{
    DeltaTime = DateTime.Now - T1;
    DroneData data = droneControl.NavigationData;

    lblVX.Text = "vX: " + Math.Round(Convert.ToDouble(data.vX), 2).ToString();
    lblVY.Text = "vY: " + Math.Round(Convert.ToDouble(data.vY), 2).ToString();
    lblVZ.Text = "vZ: " + Math.Round(Convert.ToDouble(data.vZ), 2).ToString();

    clsARDroneGeneralParams.clsPitch = Math.Round(Convert.ToDouble(data.theta), 1);
    clsARDroneGeneralParams.clsRoll = Math.Round(Convert.ToDouble(data.phi), 1);
    clsARDroneGeneralParams.clsintAltitude = data.altitude;
    clsARDroneGeneralParams.clsYaw = Math.Round(Convert.ToDouble(data.psi), 1);
```

En la anterior imagen se muestra tan solo una parte de los parámetros y valores que se escriben en la interfaz gráfica y en clsARDroneGeneralParams, llegados a este punto ya tendremos disponibles todos los valores de vuelo que emplearemos a continuación en el sistema de vuelo y demás funcionalidades del sistema.

7.2.4 IMPLEMENTACIÓN DEL SISTEMA DE VUELO AUTÓNOMO

En este apartado se explicará el algoritmo producto del diseño anteriormente realizado para realizar un sistema eficiente de vuelo autónomo a través de una trayectoria punto a punto sobre el eje XY y giro sobre el eje vertical imaginário del dron.

7.2.4.1 VARIABLES

Por ello en primer lugar es importante definir todas las variables que hemos empleado en la implementación de dicho algoritmo.

i) distancia[i]: se trata de la distancia estimada que hay entre dos puntos de coordenadas XY, es el resultado de calcular la recta que los separa.

ii) tiempoflying[i]: se tarat del tiempo estimado en que el dron estará volando entre dos puntos hasta llegar al destino.

iii) beta[i]: se trata del ángulo que hay entre dos puntos respecto al eje Y.

iv) giro[i]: el giro que realizará el dron para orientarse hacia el punto destino.

v) ControlRoll: variable que indica el valor de Roll que deberá ejecutar el dron para volar hacia el destino.

vi) ControlPitch: variable que indica el valor de Pitch que deberá ejecutar el dron para volar hacia el destino.

vii) ControlGaz: variable que indica el valor de Gaz que deberá ejecutar el dron para volar hacia el destino.

viii) ControlYaw: variable que indica el valor de Yaw que deberá ejecutar el dron para volar hacia el destino.

ix) dVx, dVy, dH: valor de la velocidad sobre los ejes X e Y que efectúa el dron y la altura de vuelo.

x) dYaw: valor del ángulo de giro Yaw que debe efectuar el Dro en radianes.

xi) timerACC1: variable empleada como contador para avanzar entre las distintas fases de vuelo y controlar el tiempo entre cada una de ellas.

xii) Avanza: booleano para controlar si el dron se encuentra en fase de avance o de orientación.

7.2.4.2 CÁLCULO DE LAS VARIABLES

En primer lugar, la distancia punto a punto de la trayectoria se obtiene de la fórmula:

$$distancia[i] = \sqrt{(\mathrm{x[i]} - \mathrm{x[i-1]})^2 - (\mathrm{y[i]} - \mathrm{y[i-1]})^2} \text{ en metros}$$

Para estimar el tiempo de vuelo entre dos puntos, recurrimos a la fórmula de física de la velocidad V=E/T:

$$tiempoFlying = \frac{1000 * distancia[i]}{500} * 2 \text{ en segundos}$$

Para calcular el ángulo Beta:

$$beta[i] = \tan^{-1}\left(\frac{x[i]-x[i-1]}{y[i]-y[i-1]}\right) * \left(\frac{180}{\pi}\right)$$

El resto de valores para Roll, Pitch, Gaz y Yaw las obtendremos de los parámetros generales o son modificaciones de nuestro sistema de vuelo.

7.2.4.3 FUNCIÓN DE VUELO AUTÓNOMO

Como se explicó con anterioridad hemos declarado un temporizador que al activarse llama a una función llamada tmrTestControl2_Tick (), este temporizador se activa en el momento en el que el usuario pulsa sobre el botón auto de nuestra interfaz gráfica ejecutándose así el algoritmo implementado sobre esta función que vamos a analizar a continuación.

En primer lugar, se define el número de puntos que forman la trayectoria que se realizará de forma autónoma para conocer el orden de los puntos, la distancia que hay entre ellos, y el tiempo estimado entre ellos, así como el ángulo beta. Una vez recabada toda esta información obtenemos los valores de pitch, roll, yaw y gaz que empleará el dron para realizar la navegación hacia el siguiente punto, esta navegación la realizará la función Navigate (Controlroll, Controlpitch, Controlyaw, Controlgaz) que es una función propia del software LSIDrone que se encarga de realizar los giros del dron a partir de unos valores recibidos como entrada y mandar los comandos de giro al dron; como en un principio el dron se encuentra en el punto (0,0) según nuestro diseño y en tierra no realizará ningún tipo de vuelo.

Es llegado a este punto cuando comienza el bucle de control while que es el encargado de leer los puntos y determinar la acción que debe seleccionar.

Como mencionamos en la fase de diseño existen nueve casos posibles dependiendo de los puntos destinos y los puntos origen de la trayectoria, siendo una misma estructura para cada uno de los casos, pero con variaciones en los modos y valores de vuelo que va a realizar. Para una primera explicación nos centraremos en el caso 1 y posteriormente explicaremos las variaciones del resto de casos.

En primer lugar, calculamos los valores para la distancia, tiempo y giro hacia el siguiente punto de manera general.

A continuación, mediante una serie de condiciones se decidirá en que caso nos encontramos. Una vez entramos dentro del caso particular, en este caso el uno, comienza la fase de orientación. Previamente hay que explicar que se ha definido un contador timerACC1 que aumenta cada vez que se activa la función de vuelo autónomo, de esta manera controlamos que cada vez que aumenta dicho contado ha pasado una cantidad determinada de tiempo, por cuestiones de diseño nuestro temporizador se activa cada 500 milisegundos por lo que cada aumento de contador ocupa ese espacio de tiempo.

Esta fase de orientación comienza tras haber realizado los cálculos previos en un intervalo de tiempo general para todos los casos en el que el contador está comprendido

entre 8 y 12 (comienza a los 4 segundos y termina a los 6). Para el caso 1 se establece un altura dH=1000, es decir un metro, y una velocidad dVx, dVy =0 ya que estamos en fase de orientación; el giro que realizará será el calculado anteriormente en este caso, 90 grados – beta (giro[i] = 90.0f - beta[i]), y se le resta la diferencia de giro (diferenciagiro), esta variable se refiere a la diferencia entre el ángulo de giro que ya existía sobre el dron en alguna fase previa de la trayectoria; el motivo de esta resta es que siempre partimos en la orientación del ángulo de giro 0, no acumulándose los giros anteriores. Al finalizar esta serie de cálculo se vuelve a activar la función en la que nos encontramos para poder llamar a la función Navigate () y efectuar los giros.

Una vez finalizada esta fase entramos en fase Hover, en la que el valor del contador se encuentra entre 12 y 16 (6 segundos y 8), en esta fase activamos la señal para el sistema de extracción, a partir de este momento el sistema empezará a recabar datos, este sistema será explicado en el siguiente punto y funcionará en paralelo al sistema de vuelo autónomo; además ejecutamos el comando EnterHoverMode () y esperamos hasta poder acceder a la siguiente fase de vuelo.

En este momento entramos en la tercera fase, la fase de avance según nuestro diseño. En esta fase se accede al pasar los 8 segundos y finaliza cuando se cumple el tiempo estimado calculado anteriormente para la trayectoria entre los dos puntos. Llegado a este punto la variable avance es verdadera y afectará a nuestro sistema de extracción de datos, y llamamos al comando de avance del dron, btnForwardAuto.PerformClick(). Al llamar a este comando se activa un evento:

```
this.btnForwardAuto.Click += new System.EventHandler(this.btnForwardAuto_Click);
```

Este evento ejecuta la función btnForwardAuto_Click explicada en el diseño de la interfaz gráfica que establece un valor de avance a una velocidad constante de 50 m/s.

```
dVx = -500.0f + dVxCorrecion;
```

La constante dVxCorrecion se explicará en el sistema de control. Se sale del modo hover y establece una altura dH=1000.Al finalizar este proceso activamos de nuevo el temporizador de la función de vuelo autónomo.

Cuando finalizamos esta última fase avanzamos hacia el siguiente punto de la trayectoria y en el caso de ser el último punto se finaliza la navegación. Esto como hemos comentado sucede para todos los casos, a continuación, se muestra una imagen de parte del código para el algoritmo, en concreto para el caso 1.

```
        ************9 COMBINACIONES POSIBLES**********************//
    ASO 1
    (x[i] < x[i - 1] && y[i] > y[i - 1])

        timerACC1++;
        giro[i] = 90.0f - beta[i];
        labelgiro.Text = giro[i].ToString();

        //FASE ORIENTACIÓN
        if (timerACC1 >= 8 && timerACC1 < 12)
        {
            LeaveHoverMode();
            dH = 1000;
            dVx = 0.0f; dVy = 0.0f;
            dYaw = giro[i] - diferenciagiro;
            giroDefinitivo = dYaw;
            UpdateUIAsync("Girando caso 1...");
            boolpitch = false;
            boolroll = false;
            boolYaw = true;
            avanza = false;

            tmrTestControl2.Start();
        }

            //FASE HOVER
        else if (timerACC1 >= 12 && timerACC1 < 16)
        {
            tmrControl2.Start();
            tiempo1 = DateTime.Now;
            EnterHoverMode();
          // FASE DE AVANCE
    else if (timerACC1 >= 16 && timerACC1 < (16 + tiempoflying[i]))
    {
        avanza = true;

        btnForwardAuto.PerformClick();


    }
        //FIN TRAYECTORIA
    else if (timerACC1 >= (16 + tiempoflying[i]))
    {
        avanza = false;

        EnterHoverMode();

        goto sumamospunto;
    }
```

Para cada uno de los casos diferentes explicados en la fase de diseño los giros que se realizan son los siguientes (ver Estructura del algoritmo de vuelo autónomo sección de casos de trayectoria)

Tabla 6: Casos del sistema de vuelo autónomo implementación

Caso	Giro
Caso 1	$dYaw=giro[i]=90-beta[i]$
Caso 2	$dYaw=giro[i]=90-beta[i]$
Caso 3	$dYaw=giro[i]=-90-beta[i]$
Caso 4	$dYaw=giro[i]=-90-beta[i]$
Caso 5	$dYaw=giro[i]=0$
Caso 6	$dYaw=giro[i]=+180$
Caso 7	$dYaw=giro[i]=0$
Caso 8	$dYaw=giro[i]=-90$
Caso 9	$dYaw=giro[i]=+90$

Como se puede comprobar el sistema de vuelo, tiene algunas limitaciones ya que no tiene en cuenta los posibles cambios en el entorno ni errores en el giro al ser giros muy predefinidos, por ello para realizar un sistema de vuelo robusto y eficaz es necesario incorporar un sistema de control a e este sistema, por ello se han implementado las dos siguientes funcionalidades ya analizadas en la fase de diseño: Sistema de extracción de datos y Sistema de Control.

7.2.5 IMPLEMENTACIÓN DEL SISTEMA DE OBTENCIÓN DE DATOS

Para mejorar nuestro sistema de vuelo autónomo e implementar un sistema de control que realice unas correcciones durante el vuelo, es necesario saber constantemente los datos de vuelo y las posiciones que va recorriendo, así como la tasa de error de cada variable, por ello es muy importante realizar una extracción en paralelo con el vuelo del dron de los datos de interés. En el anterior punto se menciona que en el momento que hemos finalizado la primera fase de orientación se activa la señal del sistema de obtención de datos, esta señal llama a la función tmrObtainActualValues_Tick().

En esta función se realizan los cálculos necesarios para la obtención de los datos, en primer lugar, debe obtenerse el tiempo que ha tardado desde la última extracción de datos, este tiempo será muy necesario más adelante; para ello se obtiene la fecha en el momento de inicio del sistema de vuelo y al acceder a la función de obtención de datos realizamos una nueva obtención de la fecha, al realizar la diferencia obtendremos el tiempo transcurrido:

```
total = new TimeSpan(tiempo2.Ticks - tiempo1.Ticks);

double processTime = total.Milliseconds;
```

Este tiempo se encuentra en milisegundos, pero por motivos de comodidad al tratar los datos, todos los valores estarán en metros y en segundos según el Sistema Métrico Decimal. A continuación, comprobamos si nos encontramos en la fase de avance o en la fase de orientación o hover, de tal forma que se accederá a la extracción de los datos en el caso de estar en fase de avance en caso contrario tan solo iremos almacenando los valores de giro y del tiempo transcurrido, así como el anterior punto de la trayectoria; comprobar si estamos en una fase u otra se realiza a través de la variable booleana “avanza”.

En el caso de encontrarnos en la fase de avance crearemos dos vectores de valores, uno para los valores reales y otro para los ideales.

7.2.5.1 CÁLCULO DE LOS VALORES IDEALES DE VUELO

- Posición X ideal, posXideal, realizamos un cálculo matemático basado en la fórmula de la distancia recorrida**, espacio = velocidad * tiempo**.

  Hay que tener en cuenta dos casos, que el punto destino esté más lejos que el origen o por el contrario que el punto destino este antes, por lo que se tratará de un retroceso, contemplando estos dos casos tenemos que:

$\frac{processTime}{1000}$ obtenemos el tiempo en segundos desde el anterior punto de la trayectoria obtenido.

Para la velocidad debemos obtener la distancia ideal entre el punto inicial y el final y dividirlo entre el tiempo estimado en recorrer los dos puntos,

$$\frac{(\text{Math.Abs}(x[i] - x[i-1]))}{\text{tiempoflying}[i]}$$

Finalmente sumamos la distancia ya recorrida y obtenemos así la fórmula:

Para `x [i] > x [i - 1]`

$$posXideal = posXideal + \left(\frac{processTime}{1000} * \frac{(\text{Math.Abs}(x[i] - x[i-1]))}{\text{tiempoflying}[i]}\right)$$

Para `x [i] < x [i - 1]`

$$posXideal = posXideal - \left(\frac{processTime}{1000} * \frac{(\text{Math.Abs}(x[i] - x[i-1]))}{\text{tiempoflying}[i]}\right)$$

- Posición Y ideal, posYideal, se emplea una ecuación similar que para la posición x ideal, pero empleando las coordenadas de los puntos Y inicial e Y final:

Para `y [i] > y [i - 1]`

$$posYideal = posYideal + \left(\frac{processTime}{1000} * \frac{(\text{Math.Abs}(y[i] - y[i-1]))}{\text{tiempoflying}[i]}\right)$$

Para `y [i] < y [i - 1]`

$$posYideal = posYideal - \left(\frac{processTime}{1000} * \frac{(\text{Math.Abs}(y[i] - y[i-1]))}{\text{tiempoflying}[i]}\right)$$

- Velocidades de los vectores X e Y ideales, la fórmula se extrae de la anterior fórmula:

$$velYideal = \frac{(\text{Math.Abs}(y[i] - y[i-1]))}{\text{tiempoflying}[i]}$$

$$velXideal = \frac{(\text{Math.Abs}(x[i] - x[i-1]))}{\text{tiempoflying}[i]}$$

Estas variables se almacenan en el vector ideales[i] finalmente.

7.2.5.2 CÁLCULO DE LOS VALORES REALES DE VUELO

Estos valores hacen referencia, a los valores que está obteniendo el Dron en tiempo real, es decir aquellos puntos por los que sí está pasando, así como su giro y velocidades.

- Velocidades X e Y reales, se obtienen de los sensores de vuelo del dron convertidos a m/s:

$$velXreal = \frac{dVx}{1000}; \qquad velYreal = \frac{dVy}{1000};$$

- Posición X real, Xreales, se emplea mediante las fórmulas de la obtención de las coordenadas cartesianas a partir de un punto del plano:

$$Xreales += \text{Math.Cos}(\text{anguloReal}) * \frac{-\text{dVx}}{1000} * \frac{\text{processTime}}{1000}$$

- Posición Y real, Yreales, la fórmula es similar a la obtención de la coordenada X, pero emplea el coseno ya que queremos obtener la coordenada Y:

$$Yreales += \text{Math.Sin}(\text{anguloReal}) * \frac{-\text{dVx}}{1000} * \frac{\text{processTime}}{1000}$$

- El ángulo de giro, lo hemos obtenido durante la fase de orientación y lo hemos almacenado en la variable anguloReal, este ángulo se obtiene de la diferencia entre el giro Yaw y el giro realizado de la anterior trayectoria ya que como se explicó con anterioridad siempre partimos al girar del ángulo 0:

anguloReal = dYaw + diferenciagiro;

Todos estos valores son almacenados en el vector de valores reales, reales[i].

Un ejemplo de dos instancias de extracción sería la siguiente:

Tabla 7:Ejemplo de extracción

X Real	X ideal	Y real	Y ideal	X real	X ideal	Giro Real
-0,0295	0	0,05900378	0,033	0,5	0,25	90
0,15692	0	0,05900378	0,12625	0,5	0,25	0

Al finalizar en cada instante la extracción de datos se activa la señal del sistema de control que empleara dichos valores para ejecutar el sistema de reglas para realizar las correcciones durante el vuelo del dron.

7.5.3 IMPLEMENTACIÓN DEL SISTEMA DE CONTROL

El Sistema de control consiste en una función formada por un sistema de reglas con una entrada y una salida determinada para cada caso, fruto de una fase previa de diseño implantación y pruebas, seleccionando aquellos casos en los que se reduce de mayor forma el error durante una fase de vuelo.

Este sistema recibe los dos vectores de valores previamente mencionados y los procesa para determinar a qué determinado caso del sistema de reglas puede acceder. Tras haber realizado un análisis previo se ha comprobado que los errores se producen principalmente en la efectuación de los giros del dron para orientarse hacia una determinada posición y a la velocidad que está establecida a 50 m/s como constante, por ello este sistema de control de reglas emplea un total de 19 casos posibles en los que esté enmarcada la fase vuelo.

En primer lugar, se han implementado tres reglas para realizar correcciones sobre la velocidad del eje X, como ya sabemos por diseño el dron cuando realiza un proceso de avance emplea una velocidad de signo negativo, y de signo negativo en caso de retroceder. Una vez contemplado esto, realizamos una comparación entre la

velocidad real y la velocidad ideal, produciéndose una entrada en cualquiera de los tres casos, y ejecutándose la regla de corrección de velocidad:

```
//Correcciones X
if (velXreal > velXideal)
{
    //Para relentizar sumar velocidad a dVx
    dVxCorrecion +=20.0f;
}
if (velXreal < velXideal)
{
    //Para acelerar restar velocidad a dVx
    dVxCorrecion -= 20.0f;
}
if (velXreal < 0.15)
{
    //Para acelerar restar velocidad a dVx
    dVxCorrecion -= 10.0f;
}
```

Con estas reglas comprobamos si se excede o si la velocidad no es suficiente respecto a la velocidad que debiese llevar, además la tercera regla evita llevar a cabo desaceleraciones bruscas. La variable dVxCorrecion, se trata de la variable que cuando el dron se encuentra en fase de avance, para la siguiente instancia realizará la corrección de velocidad a medida que se desplaza el dron:

```
private void btnForwardAuto_Click(object sender, EventArgs
{
    //Desplazamiento hacia delante(avance)
    LeaveHoverMode();
    dH = 1000;
    dVx = -500.0f + dVxCorrecion;
```

A continuación, están implementadas las reglas encargadas de corregir el giro del dron dependiendo de la localización de las coordenadas reales respecto de las coordenadas ideales.

Tabla 8:Reglas del sistema de control

Regla 4	`reales[0] < ideales[0] && reales[1] == ideales[1]`
Regla 5	`reales[0] == ideales[0] && reales[1] < ideales[1]`
Regla 6	`reales[0] > ideales[0] && reales[1] == ideales[1]`
Regla 7	`reales[0] == ideales[0] && reales[1] > ideales[1]`
Regla 8	`reales[0] < ideales[0] && reales[1] < ideales[1]`
Regla 9	`reales[0] > ideales[0] && reales[1] > ideales[1]`
Regla 10	`reales[0] > ideales[0] && reales[1] < ideales[1]`
Regla 11	`reales[0] < ideales[0] && reales[1] > ideales[1]`

- Regla 4, se activa en el caso que la coordenada Real X se encuentre a la izquierda de la coordenada Ideal, entonces realiza un giro de orientación 0 grados y llama a la función de navegación para efectuarlo.
- Regla 5, se activa en el caso de que la coordenada Real Y sea menor que la coordenada Ideal, entonces realiza un giro a la izquierda de 90 grados y llama a la función de navegación.
- Regla 6, se activa en el caso que la coordenada Real X se encuentre a la derecha de la coordenada Ideal, entonces realiza un gran giro hacia la izquierda de orientación 180 grados y llama a la función de navegación para efectuarlo.
- Regla 7, se activa en el caso de que la coordenada Real Y sea mayor que la coordenada Ideal, entonces realiza un giro a la izquierda de 90 grados y provoca una deceleración en el dron, finalmente llama a la función de navegación.
- Regla 8, se activa en el caso de que la coordenada Real Y sea menor que la coordenada Ideal Y, y la coordenada Real X sea menor que la coordenada Ideal X, entonces realiza un giro a la derecha de orientación 0 grados y provoca una aceleración en el dron, finalmente llama a la función de navegación.

- Regla 9, se activa en el caso de que la coordenada Real Y sea mayor que la coordenada Ideal Y, y la coordenada Real X sea mayor que la coordenada Ideal X, entonces realiza un giro a la izquierda de orientación 180 grados y provoca una deceleración en el dron, finalmente llama a la función de navegación.

- Regla 10, se activa en el caso de que la coordenada Real Y sea menor que la coordenada Ideal Y, y la coordenada Real X sea mayor que la coordenada Ideal X, entonces realiza un giro a la izquierda de orientación 90 grados y finalmente llama a la función de navegación.

- Regla 11, se activa en el caso de que la coordenada Real Y sea mayor que la coordenada Ideal Y, y la coordenada Real X sea menor que la coordenada Ideal X, entonces realiza un giro a la derecha de orientación 0 grados y finalmente llama a la función de navegación.

```
    eales[0] < ideales[0] && reales[1] == ideales[1])

    UpdateUIAsync("Regla 4: Corrige derecha...");
    //Giro derecha
    anguloReal = 0;
    Navigate(Controlroll, Controlpitch, 0, Controlgaz);

}
if (reales[0] == ideales[0] && reales[1] < ideales[1])
{
    UpdateUIAsync("Regla 5:Corrige izda y acelera...");
    //Giro izda
    anguloReal = 90;
    dVxCorrecion -= 10.0f;
}
if (reales[0] > ideales[0] && reales[1] == ideales[1])
{
    UpdateUIAsync("Regla 6 : Corrige izda...");
    //Giro iquierda brusco
    anguloReal = 180;
    Navigate(Controlroll, Controlpitch, 180, Controlgaz);

}
if (reales[0] == ideales[0] && reales[1] > ideales[1])
{
    UpdateUIAsync("Regla 7: Corrige derecha y frena...");
    //Giro derecha
    //Deceleramos
    anguloReal = 90;
    dVxCorrecion += 10.0f;
    Navigate(Controlroll, Controlpitch, 0, Controlgaz);

}
if (reales[0] < ideales[0] && reales[1] < ideales[1])
{
    UpdateUIAsync("Regla 8 :Corrige izda y acelera...");
    //Giro derecha
    //Aceleramos
    anguloReal = 0;
    dVxCorrecion -= 10.0f;
    Navigate(Controlroll, Controlpitch, 0, Controlgaz);

}
if (reales[0] > ideales[0] && reales[1] > ideales[1])
{
    UpdateUIAsync("Regla 9 :Corrige derecha y frena...");
    //Giro izquierda brusco
    //Deceleramos
    anguloReal = 180;
    dVxCorrecion += 10.0f;
    Navigate(Controlroll, Controlpitch, 180, Controlgaz);

}
if (reales[0] > ideales[0] && reales[1] < ideales[1])
{
    UpdateUIAsync("Regla 10:Corrige izquierda y acelera...")

    //Giro izquierda

    anguloReal = 90;
    Navigate(Controlroll, Controlpitch, 90, Controlgaz);


}
if (reales[0] < ideales[0] && reales[1] > ideales[1])
{
    UpdateUIAsync("Regla 11 :Corrige a la derecha...");
    //Giro dcha
    anguloReal = 0;
    Navigate(Controlroll, Controlpitch, 0, Controlgaz);
```

Ilustración 35:Código de Reglas Sistema autónomo

7.5.4 IMPLANTACIÓN DEL PROTOCOLO MAVLINK

Tras la implantación del sistema de control, hemos conseguido obtener un sistema de vuelo autónomo eficiente que es capaz de realizar correcciones en tiempo real durante una trayectoria punto a punto, consiguiendo así el sistema de control de vuelo autónomo sobre cuadricóptero objetivo de este proyecto.

No obstante, con fines de investigación se ha decidido implantar el protocolo MavLink en nuestro Dron a través de la interfaz del Software QgroundControl para obtener más información y realizar comparaciones de trayectorias de vuelo respecto nuestro sistema autónomo y consiguiendo así una alternativa de control de vuelo.

7.5.4.1 CABECERA MAVLINK

Lo primero que debemos obtener para implantar este sistema es la cabecera MavLink que debe implementarse en QgroundControl con las opciones y parámetros que queremos obtener durante el vuelo que programemos en el software.

Para ello se debe de emplear el Software MavLink Generate, se trata de un programa en Python que recibe como entrada la definición XML del mensaje MavLink que queremos obtener y nos generará la cabecera en el lenguaje deseado, ya sea C, C# o Python, en nuestro caso será c#.

El mensaje en XML que ha sido creado está formado por las siguientes etiquetas que serán reconocidas por nuestro software MavLink Master para generar la cabecera:

- MavLink Protocol Version. Indica la versión de MavLink que queremos generar en nuestro caso, 1.0v.
- MAV_AUTOPILOT, indica la versión del UAV empleado para este protocolo MAV. MAV_AUTOPILOT_GENERIC en este caso.
- MAV_TYPE, indica el tipo de UAV en nuestro caso MAV_TYPE_QUADROTOR.
- FIRMWARE_VERSION_TYPE, indica la versión del firmware de MavLink, en este caso, FIRMWARE_VERSION_TYPE_DEV, la versión para desarrolladores.

- MAV_MODE_FLAG, indica los flags que codifican el modo del MAV, se han introducido todos los tipos de modo, para poder alternar entre ellos si es necesario.

- MAV_MODE_FLAG_DECODE_POSITION, indica cómo se decodifica la posición del dron. Hemos incluido todos los modos de obtención de posiciones.

- MAV_GOTO, indica los comandos de movimiento que queremos comunicar a través del MAV.

- MAV_SYS_STATUS_SENSOR, establece el contenido de los sensores que queremos mandar a través del mensaje MavLink, se han establecido los valores por defecto para que se autodetecten los sensores del dron.

- MAV_FRAME, configura que tipo de sistema de referencia queremos emplear en las trayectorias del dron, es principalmente el cuerpo del mensaje. MAV_FRAME_GLOBAL.

Existen multitud de etiquetas que posibilitan muchas configuraciones, pero para el objetivo de este proyecto, estas son las principales etiquetas que debemos configurar, el resto de etiquetas empleadas se utilizan por defecto.

Una vez configurado el mensaje debemos ejecutar el software, y nos generará automáticamente la cabecera que incluir en las librerías de QgroundControl.

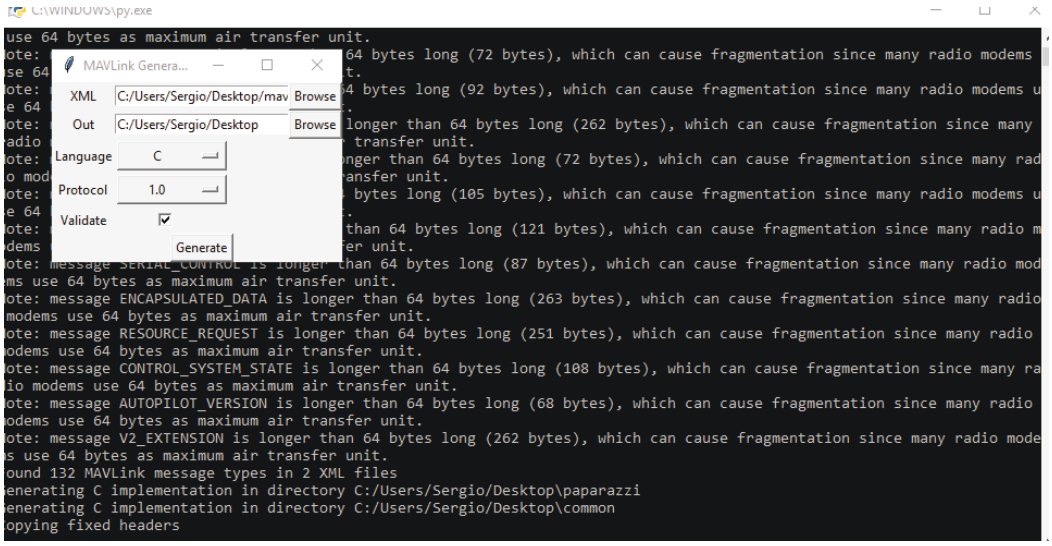


Ilustración 36:Generador de cabecera

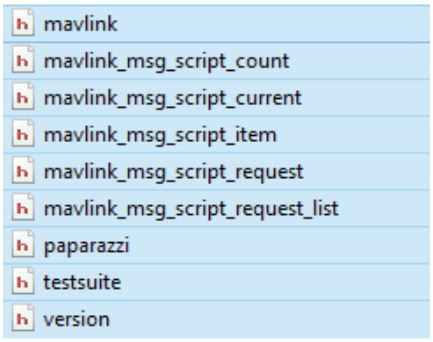


Ilustración 37: Ficheros de la cabecera generados

Una vez obtenidos estos ficheros debemos obtener nuestro software QgroundControl.

7.5.5 EXTRACCIÓN DE DATOS VÍA QGROUNDCONTROL

Para poder modificar las cabeceras personalizadas MavLink del software QgroundControl, en primer lugar, debemos obtener el código fuente del software ya que se trata de un programa OpenSource, por ello a través de los repositorios se ha descargado la "versión 2.7 Stable".

En segundo lugar, para poder ejecutar este código existen varias opciones con diferentes entornos de programación, ya sea c++ o c#, que para seguir en la línea del resto del proyecto hemos utilizado la versión del software c#. Para poder ejecutarlo es necesario primero emplear QT 5.4 para compilar el código de QgroundControl. Una vez compilado se genera un ejecutable que se depurará mediante Visual Studio 2013 Express.

Tras obtener la compilación del software y ya disponer de las cabeceras obtenidas en el anterior punto, debemos incluirlas en el proyecto compilado, para ello debemos sustituir los ficheros de las librerías de MavLink (/include/MavLink/v1.0/) del proyecto QgroundControl por nuestras cabeceras. De esta forma nuestro software permitirá la extracción de los datos que hemos indicado en las cabeceras de los mensajes Mav y la configuración correcta para realizar la comunicación con nuestro dron. Llegados a este punto ya podemos ejecutar nuestro software y ejecutar las trayectorias deseadas.

QgroundControl una vez ejecutado muestra una interfaz muy intuitiva para realizar planes de vuelo. En primer lugar, dispone de la pestaña "Plan" que nos permite ejecutar la trayectoria de vuelo deseada, que mediante el software del Parrot "GPS Fight Recorder" obtiene la posición GPS actual del dron, mostrándonos un mapa con su

localización. Una vez tenemos localizado el dron tan solo debemos marcar en el propio mapa los Waypoint que formarán la trayectoria del Dron, así como su configuración de vuelo (altura de vuelo, tipo de Waypoint…). Además, nos permite visualizar en tiempo real la información de los paquetes MavLink que se están trasmitiendo entre la estación y el dron y la posibilidad de extraer los datos que se han ido obteniendo durante el vuelo en dicha transmisión de mensajes.

En segundo lugar, dispone de la pestaña “Fly”, que muestra los parámetros e vuelo, así como la inclinación de sus ejes, es muy similar a las empleadas por los aviones en una navegación.

Y por último la pestaña “Analyze” nos permite obtener gráficas de vuelo y guardarlas teniendo como posibles valores X e Y aquellos que se obtienen a través de MavLink.

De esta forma tenemos un sistema alternativo de extracción de información de vuelo en tiempo real, a través de un protocolo de comunicaciones seguro y un software OpenSource, que nos servirá de complemento a nuestro sistema de control de vuelo autónomo.

8 DIAGRAMA FINAL DEL SISTEMA

A continuación, se muestra un diagrama de flujo del sistema de control de vuelo autónomo implementado en nuestro dron.

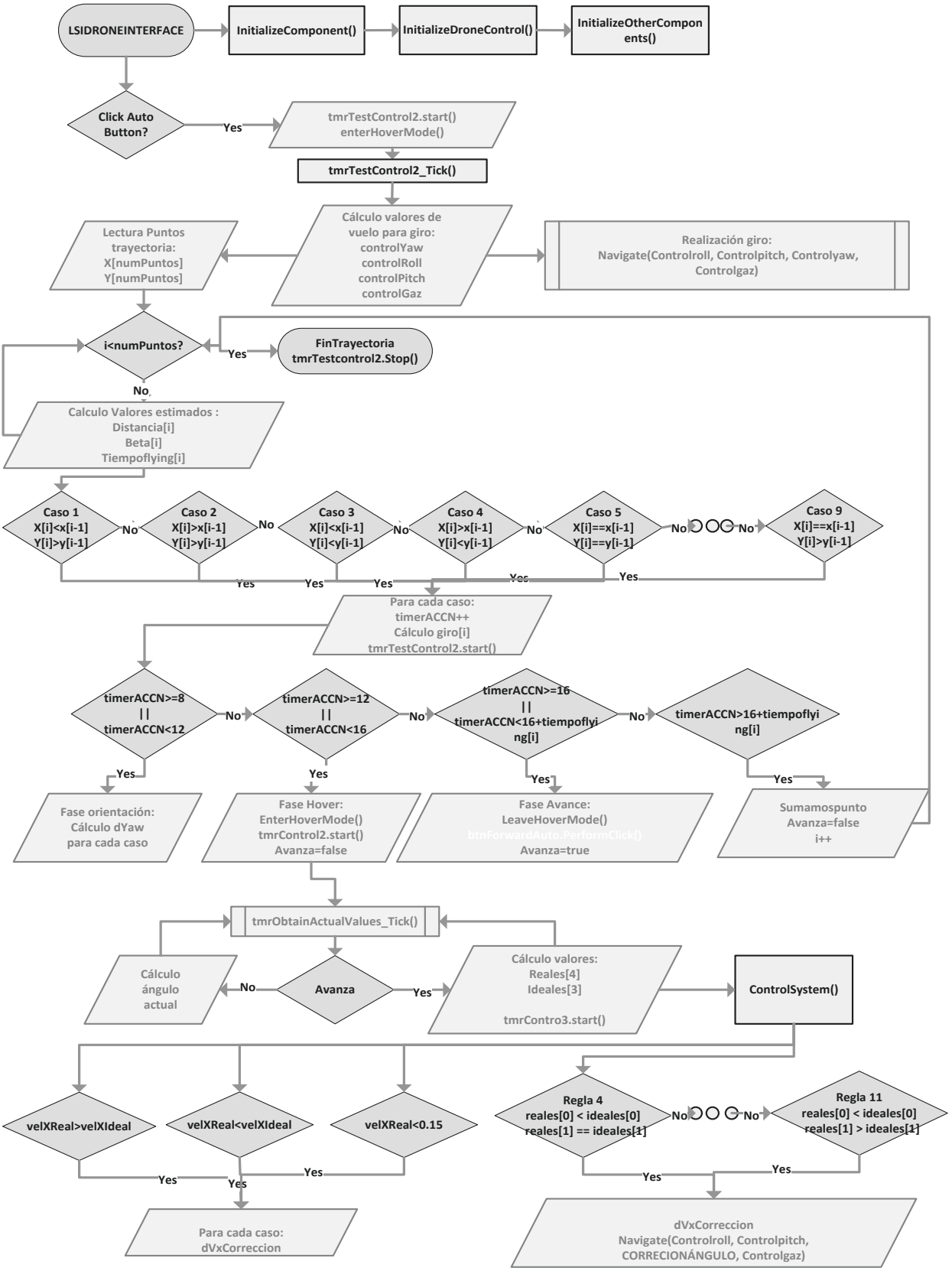


Diagrama 7: Diagrama final del sistema de control de vuelo autónomo

9 RESULTADOS Y EVALUACIÓN

En este punto se va a realizar un análisis exhaustivo de los resultados obtenidos para una serie de trayectorias ejecutadas por nuestro sistema de vuelo autónomo comparando de forma gráfica los resultados obtenidos empleando el sistema vuelo autónoma sin el sistema de control, y por otro lado empleando el sistema de control de trayectorias, permitiendo así observar la eficacia del sistema implementado en el dron y su funcionamiento.

9.1 TRAYECTORIA 1

Para esta primera trayectoria se ha establecido una ruta de vuelo formada por tres puntos, que consiste en un giro a la izquierda y posteriormente un giro a la derecha.

Trayectoria= (0,0), (0,1), (1,1)

En primer lugar, se ha realizado el vuelo autónomo sin emplear el sistema de control obteniendo tanto la trayectoria ideal del dron, y la trayectoria real o que realmente está llevando el dron. En la siguiente gráfica se representan ambas trayectorias, cabe destacar que las unidades están en metros.

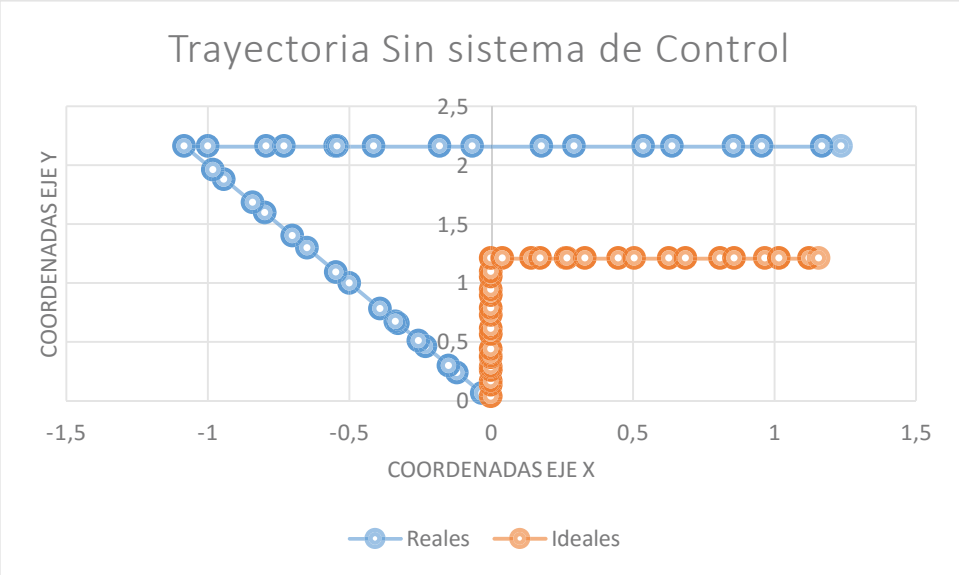


Ilustración 38: Gráfica de trayectoria sin sistema de control 1

Como se puede observar existe bastante variación entre la trayectoria que sigue el dron y la que debería estar siguiendo. Este error se debe principalmente como se observa en la gráfica a que el giro realizado no es del todo preciso sumado a una velocidad inadecuada, la causa principal de este desfase es el error en el cálculo tras la

estimación de la distancia al nuevo punto, por ello se comete un error en la velocidad además del error en el giro realizado, se mantiene más tiempo del adecuado realizando el giro en la fase de orientación sobrepasando el ángulo de giro y tras entrar en la fase de avance no es capaz de corregirlo.

A continuación, se ha realizado la misma trayectoria, pero con el sistema de control de trayectorias activado, con su consiguiente representación:

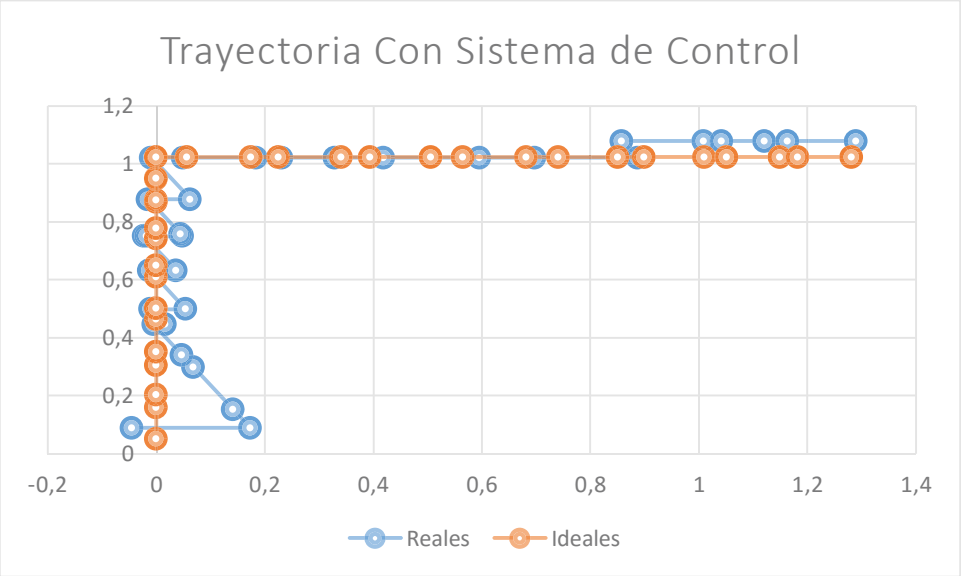


Ilustración 39: Gráfica de trayectoria con sistema de control 1

Como se puede observar la trayectoria se ajusta en gran medida a la trayectoria ideal del dron, debido a las correcciones que ejecuta el sistema de control durante el propio vuelo autónomo, como se ha explicado en la implementación realiza una serie de giros para realizar correcciones y además modifica la velocidad(m/s) como se muestra a continuación:

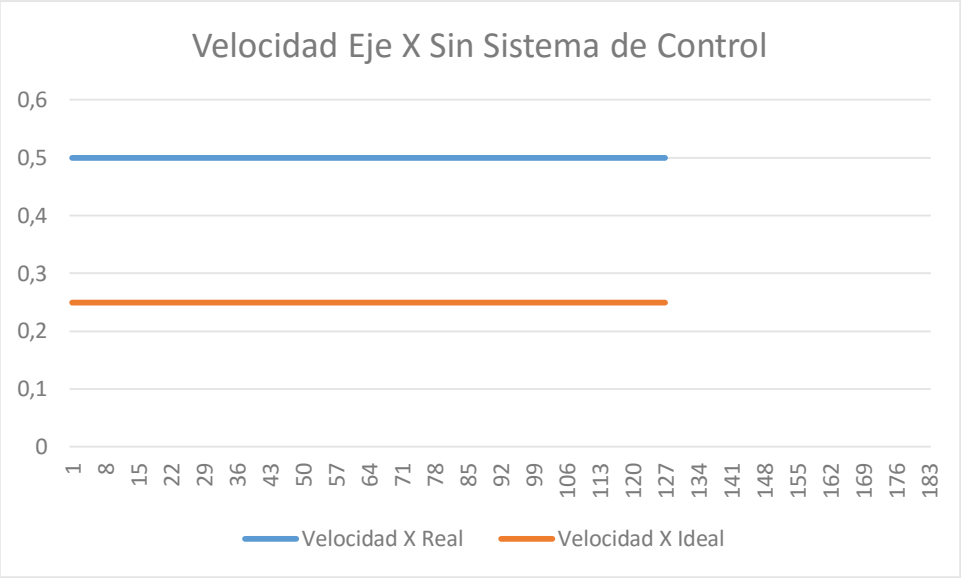


Ilustración 40:Gráfica de velocidades sin sistema de control

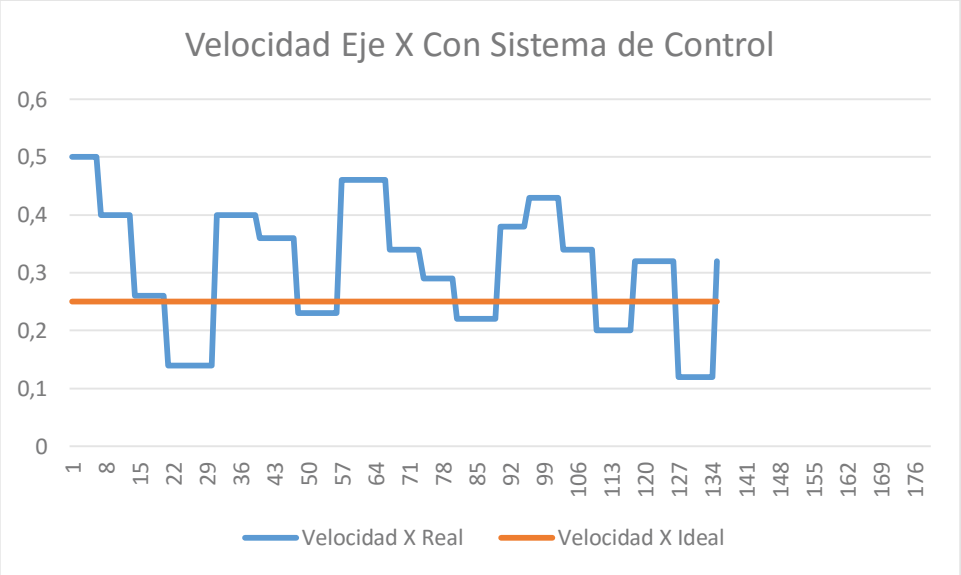


Ilustración 41: Gráfica de velocidades con sistema de control

Como se observa se realizan continuas modificaciones de la velocidad aumenta o disminuye teniendo en cuenta la diferencia con la velocidad que debería llevar, pasa de llevar una velocidad constante, a una velocidad en continua adaptación.

Durante este proceso se activan las reglas de giro de la orientación del dron correspondientes, dependiendo de la posición del dron respecto de la trayectoria ideal como se explicó con anterioridad:

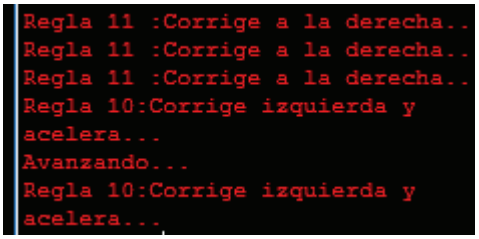


Ilustración 42: Reglas activadas del sistema de control trayectoria 1

La traza de activación de reglas de modificación de giro que lleva acabo el dron para esta trayectoria es la siguiente:

Tabla 9: Reglas activadas en la trayectoria 2

Regla 11: Corrige a la derecha...
Regla 10: Corrige izquierda y acelera...
Regla 9: Corrige derecha y frena...
Regla 10: Corrige izquierda y acelera...
Regla 8: Corrige izda y acelera...
Regla 10: Corrige izquierda y acelera...
Regla 8: Corrige izda y acelera...
Regla 10: Corrige izquierda y acelera...
Regla 8: Corrige izda y acelera...
Regla 10: Corrige izquierda y acelera...
Regla 8: Corrige izda y acelera...

Regla 10: Corrige izquierda y acelera...
Regla 8: Corrige izda y acelera...
Regla 10: Corrige izquierda y acelera...
Regla 8: Corrige izda y acelera...
Regla 10: Corrige izquierda y acelera...
Regla 8: Corrige izda y acelera...
Regla 10: Corrige izquierda y acelera...
Regla 8: Corrige izda y acelera...
Regla 10: Corrige izquierda y acelera...
Regla 8: Corrige izda y acelera...
Regla 10: Corrige izquierda y acelera...
Regla 11: Corrige a la derecha...
Regla 9: Corrige derecha y frena...
Regla 11: Corrige a la derecha...
Regla 9: Corrige derecha y frena...
Regla 11: Corrige a la derecha...
Regla 9: Corrige derecha y frena...
Regla 11: Corrige a la derecha...
¡¡FIN trayectoria control !!

Por último, se puede observar cómo se realiza una gran disminución del error medio tanto para el eje X como para el eje Y de la trayectoria del dron, realizando una

trayectoria que se ajusta prácticamente a la trayectoria que deseamos, realizándose un vuelo de forma autónoma y con la capacidad de adaptarse ante los errores de vuelo, evitando así que se vaya acumulando y aumentando el error, y consiguiendo siempre adaptar nuestro vuelo a la trayectoria correcta.

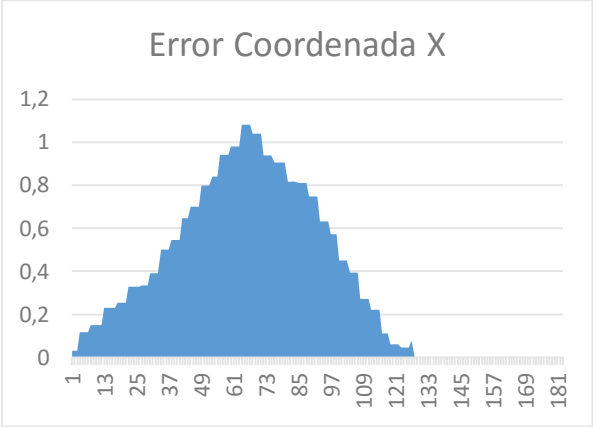


Ilustración 44: Error X sin Control Trayectoria 1

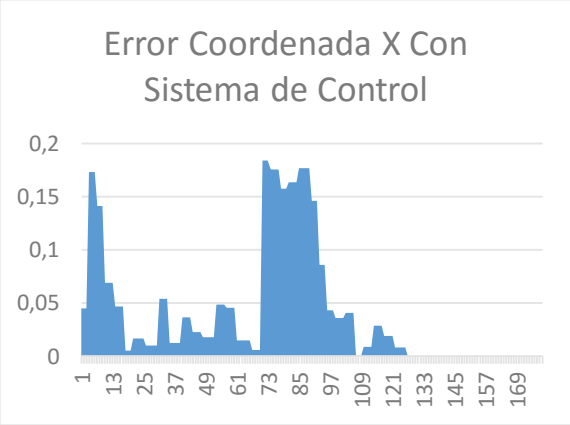


Ilustración 43: Error X con Control Trayectoria 1

Ilustración 46: Error Y sin Control Trayectoria 1

Ilustración 45: Error Y con Control Trayectoria 1

Comparativa del error medio tanto para las coordenadas X como las coordenadas Y:

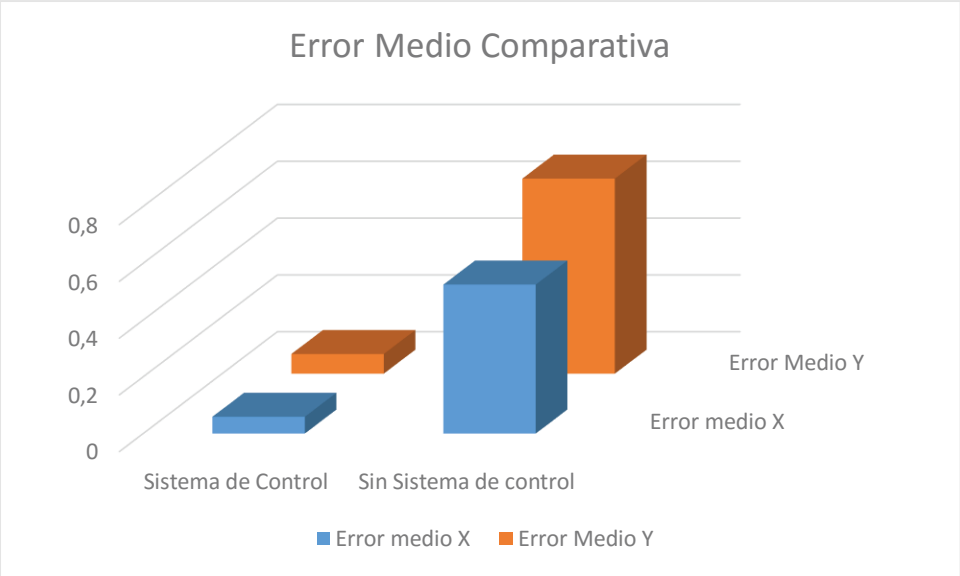


Ilustración 47: Error medio Trayectoria 1

9.2 TRAYECTORIA 2

En segundo lugar, se ha realizado una trayectoria de vuelo autónomo que está formada por cinco waypoints o puntos de paso:

Trayectoria= (0,0), (0,1), (-1,1), (-2,2), (0,2)

En primera instancia se ha realizado dicha trayectoria sin el sistema de control activado generando los siguientes resultados en su trayectoria:

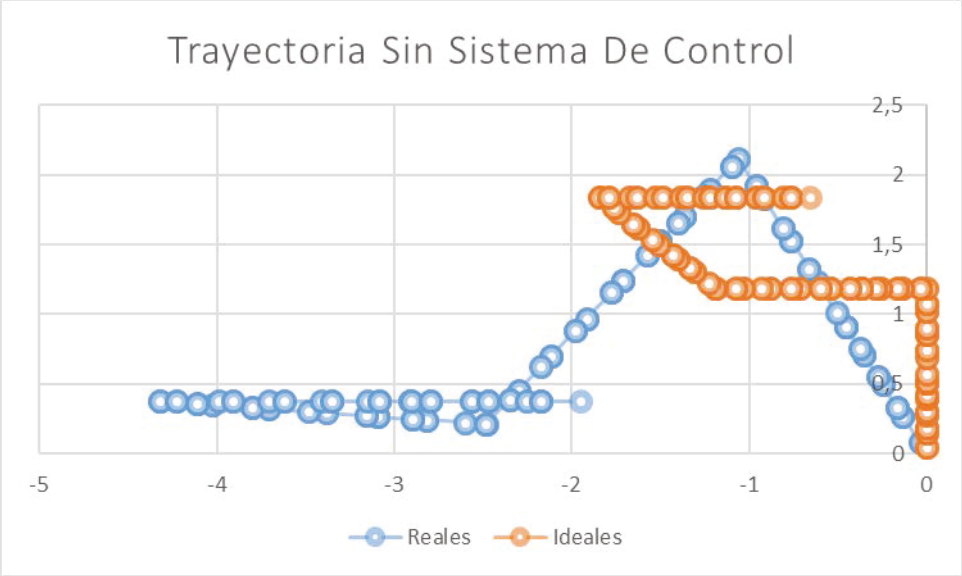


Ilustración 48: Trayectoria sin sistema de control 2

Como se puede observar, el sistema autónomo no es capaz de seguir de forma correcta los puntos de paso debido a su limitación en los giros y al empleo de una velocidad siempre constante. Esto provoca, al ser una trayectoria de poco recorrido una estimación incorrecta en el tiempo de llegada a un punto de paso debido a la velocidad excesiva, y acarre un error en la trayectoria que se va acumulando. Además, se observan giros incluso en sentido contrario, cuyo principal motivo es que el sistema de vuelo autónomo no es capaz de encontrar la orientación hacia el punto siguiente entrando en un caso de trayectoria erróneo debido al desfase de posición tan grande acarreado durante todo el recorrido.

Sin embargo, al emplear el sistema de control de vuelo autónomo los resultados mejoran de forma muy notable:

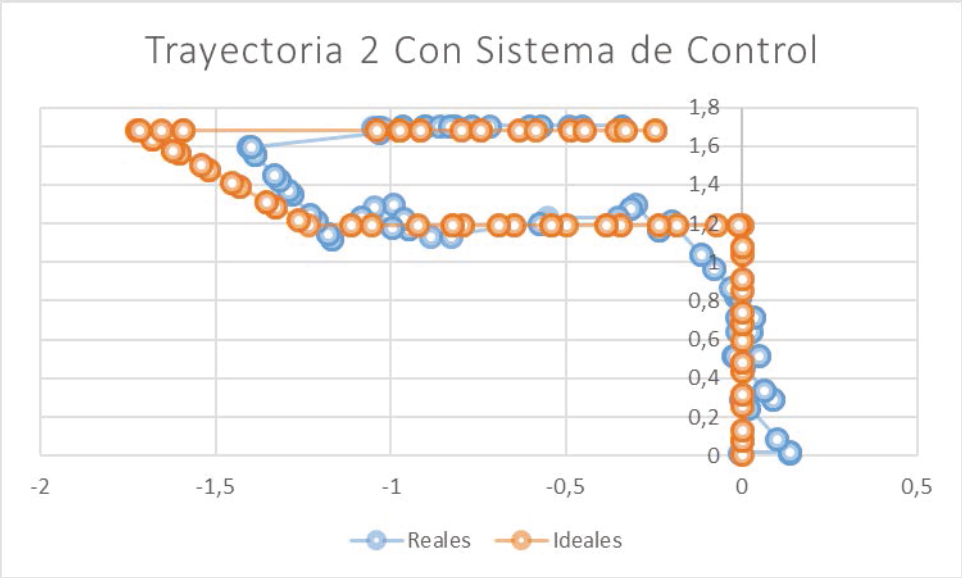


Ilustración 49:Trayectoria con sistema de control 2

La trayectoria que sigue el dron como se observa es muy similar a la trayectoria ideal, consiguiendo llegar a los puntos objetivos y realizar de forma correcta la trayectoria llegando al punto destino.

Tras analizar las trayectorias de los dos sistemas, se puede afirmar que la gran diferencia en la velocidad, así como una incorrecta orientación del dron al seguir su trayectoria se han conseguido corregir en tiempo real con este sistema, produciendo una trayectoria punto a punto que se ajusta de forma muy eficiente al recorrido ideal. A continuación, se muestran las gráficas de velocidades:

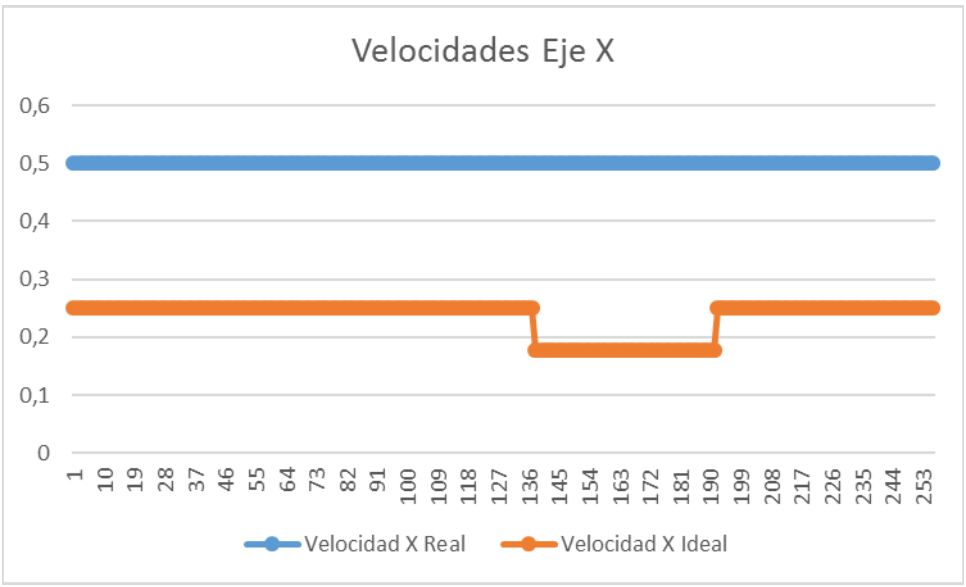


Ilustración 50:Velocidades sin sistema de control trayectoria 2

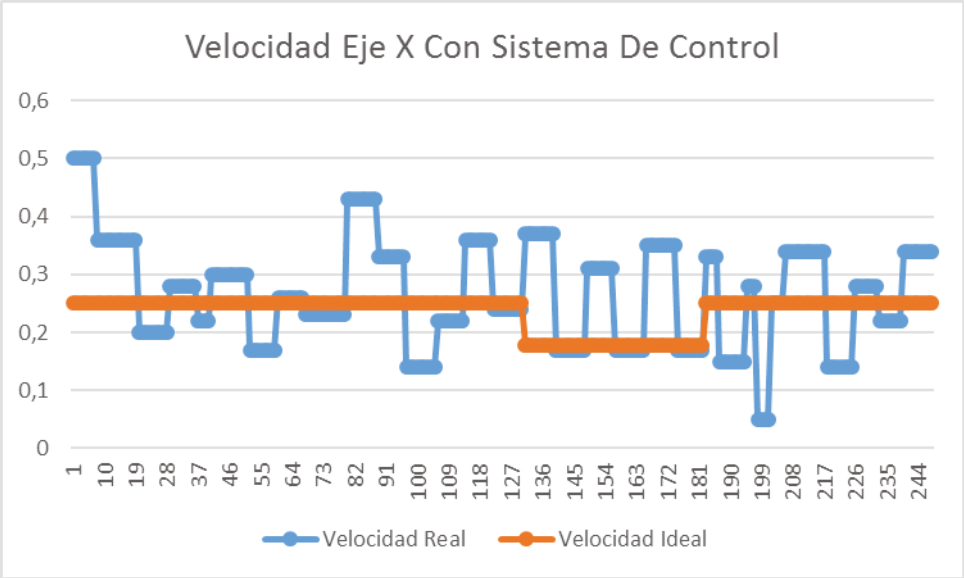
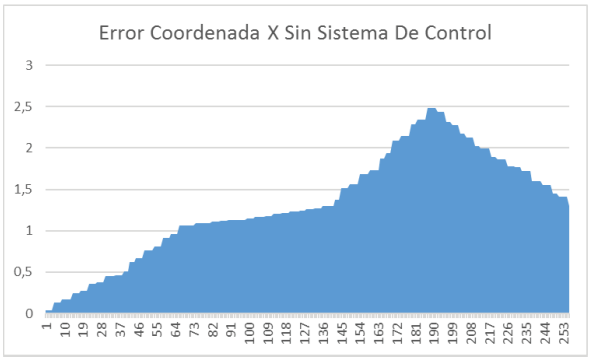


Ilustración 51:Velocidades con sistema de control trayectoria 2

Como muestran las figuras, el sistema corrige constantemente la velocidad produciendo aceleraciones y deceleraciones mientras avanza el dron hacia su objetivo tratando de conseguir la velocidad que se estima es la óptima para alcanzar en el tiempo estimado y la distancia estimada el objetivo.

Por último, se presenta una comparativa de errores en cada momento de la trayectoria, así como el error medio de cada coordenada durante todo el recorrido, observándose la optimización del error:



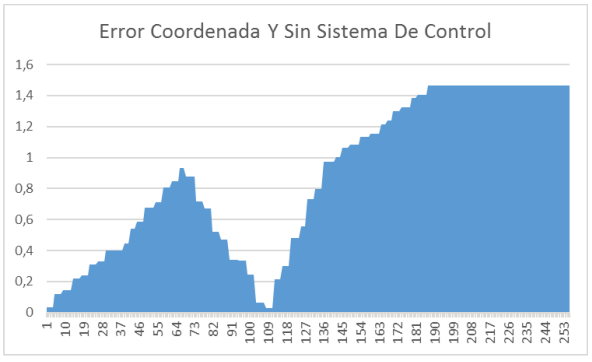


Ilustración 52:Error sin sistema de control trayectoria 2

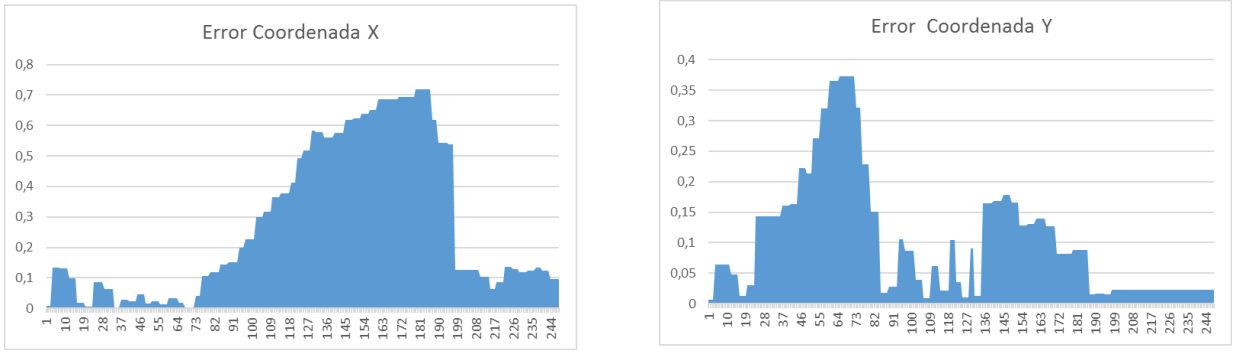


Ilustración 53: Error con sistema de control trayectoria 2

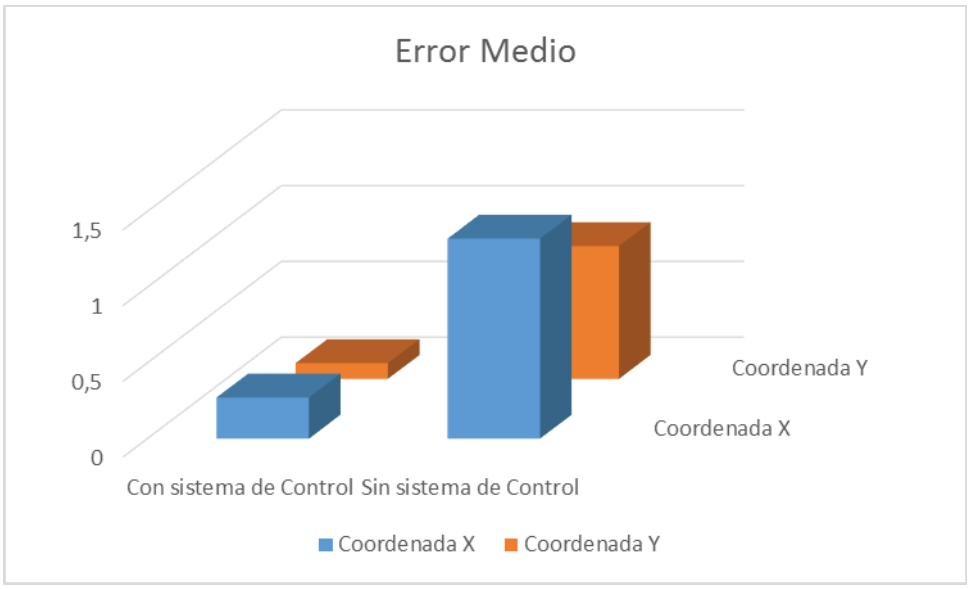


Ilustración 54:Error medio trayectoria 2

9.3 TRAYECTORIA 3

En tercer lugar, se presenta una trayectoria formada por 5 waypoints, con la que se quiere comprobar el comportamiento del dron ante cambio rápidos de orientación.

Trayectoria= (0,0), (0,1), (1,1), (1,1.5), (1.5,1.5)

Siguiendo la estructura de estudio de las trayectorias de los anteriores puntos, hemos analizado el sistema con el sistema de control activado y sin él, generando los siguientes resultados en su recorrido punto a punto:

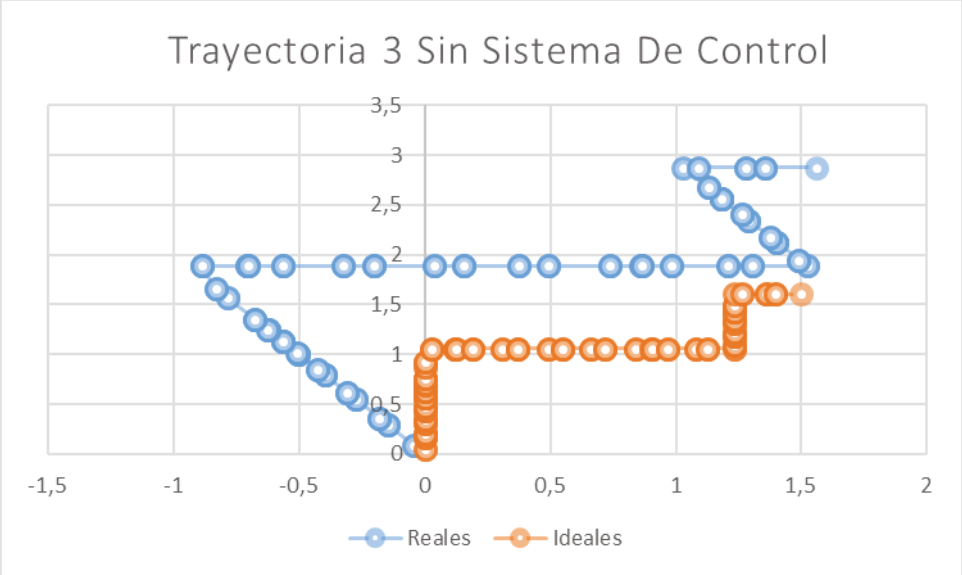


Ilustración 55:Trayectoria 3 sin sistema de control

Para el vuelo autónomo sin el sistema de control activado, se observa que esta vez sí ha estimado de forma aceptable la velocidad a emplear en el recorrido, sin embargo, en la fase de orientación del dron se han cometido errores que se han acarreado durante todo el recorrido.

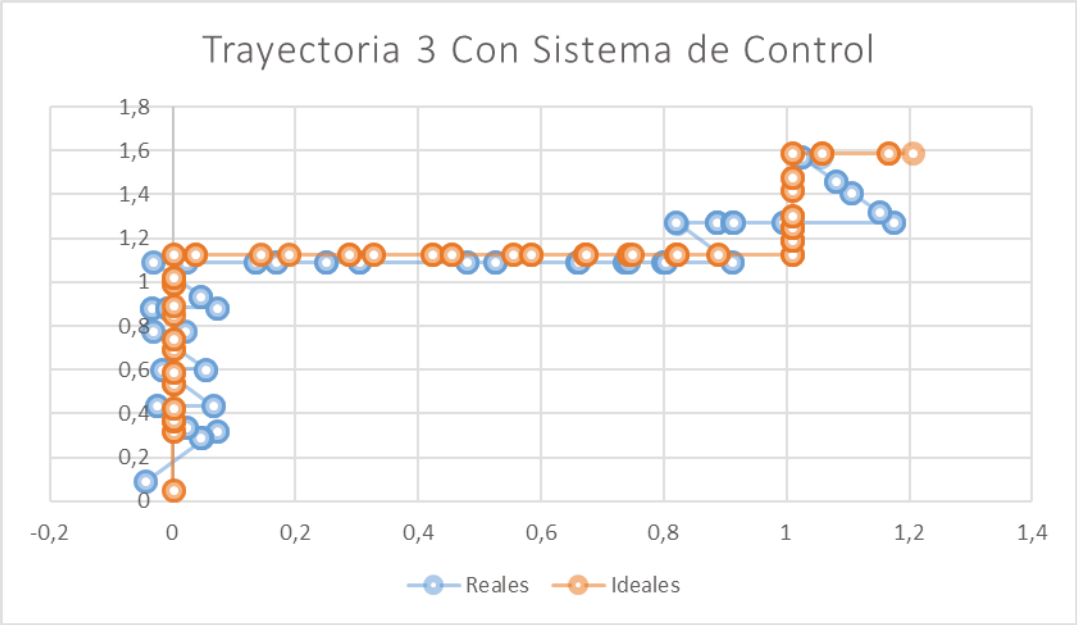


Ilustración 56: Trayectoria 3 Con sistema de control

Una vez activado el sistema de control como se muestra con la anterior figura que representa el recorrido seguido se ha conseguido nuevamente ajustar la trayectoria a la deseada. Por parte de la velocidad como se mencionó anteriormente el sistema autónomo sin el control activado, ha conseguido aproximar bastante la velocidad real a la ideal, no obstante, sigue existiendo un error bastante notable.

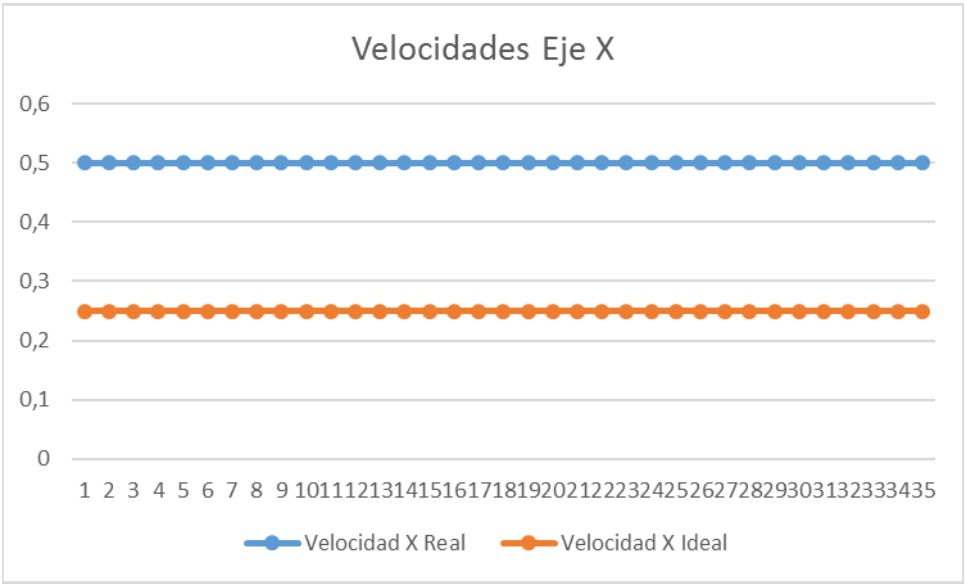


Ilustración 57:Velocidades sin sistema de control trayectoria 3

Con el sistema de control activado se obtienen los siguientes resultados para las velocidades:

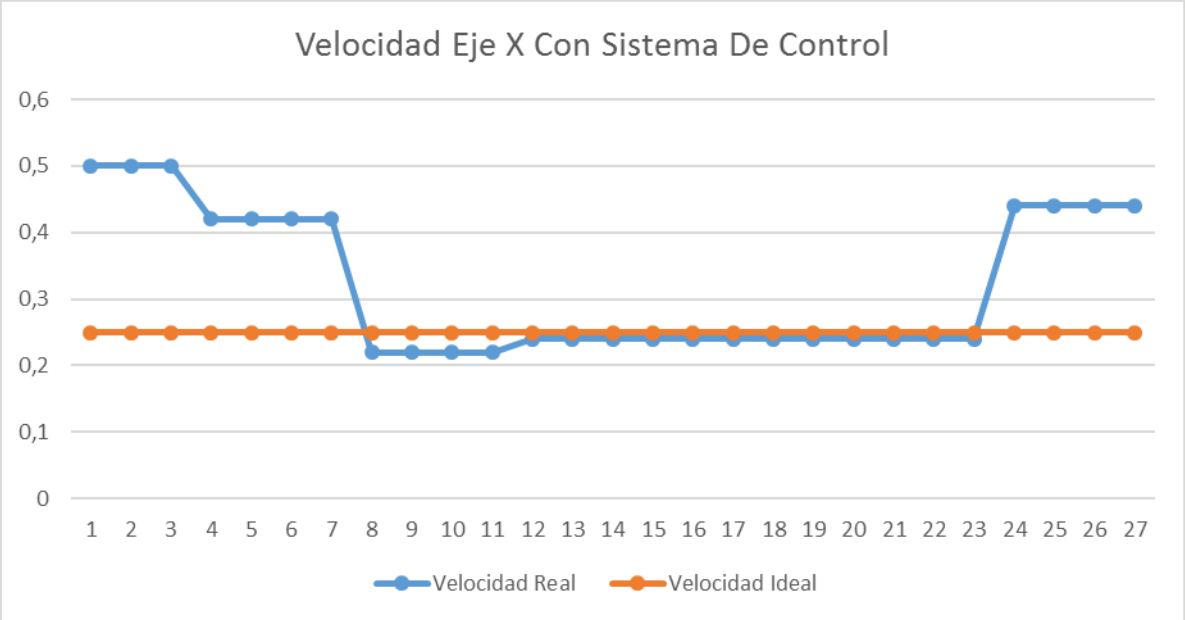


Ilustración 58:Velocidades con sistema de control trayectoria 3

Por último, se muestran los errores para la trayectoria ejecutada en cada instante de tiempo junto al error medio, mostrándose claramente que el sistema de control nuevamente consigue mejorar la trayectoria y el acarreo del error:

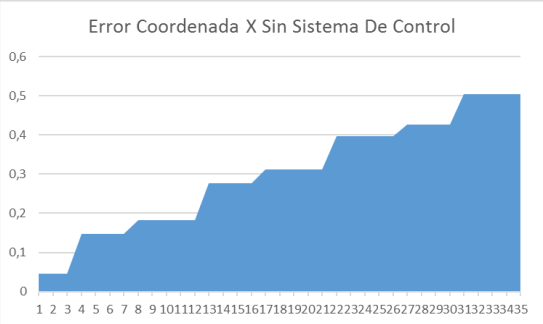


Ilustración 59:Error Coordenada X sin sistema de control

Ilustración 60:Error coordenada Y sin sistema de control

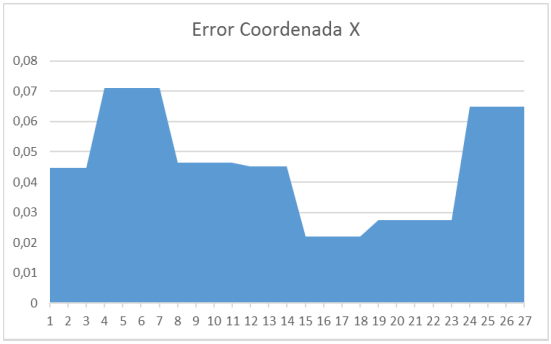


Ilustración 61:Error Coordenada X con sistema de control

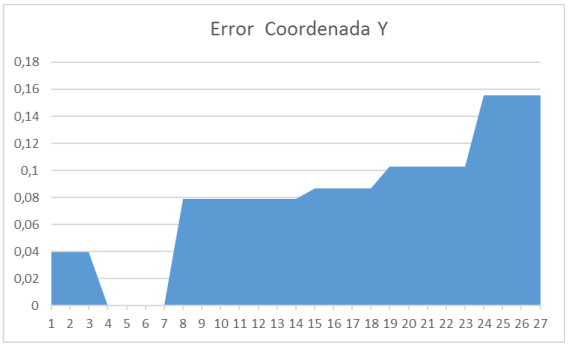


Ilustración 62:Error Coordenada Y con sistema de control

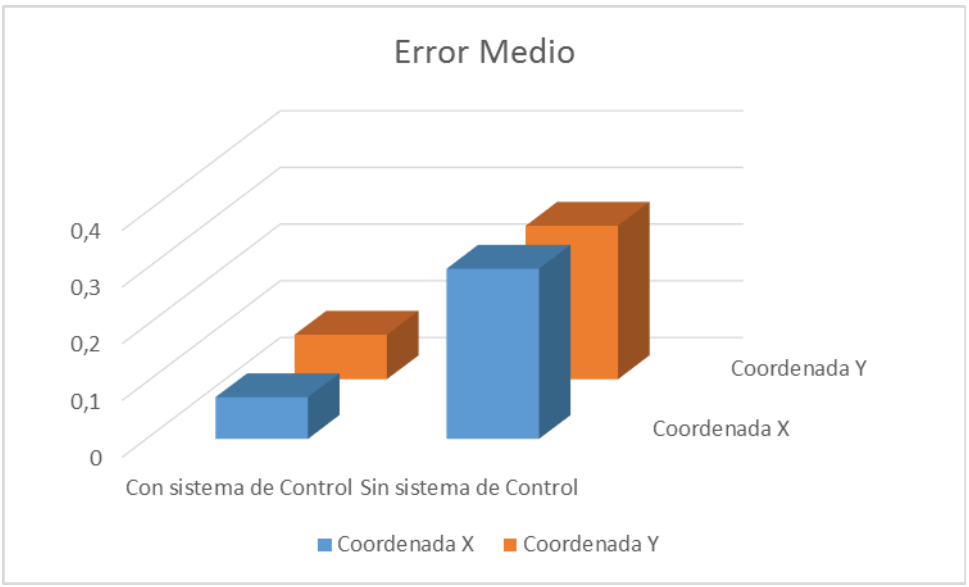


Ilustración 63:Error medio trayectoria 3

10 CONCLUSIONES FINALES

10.1 CONCLUSIONES Y OBJETIVOS CUMPLIDOS

Tras haber realizado el análisis del problema planteado, estudiado su marco histórico y entorno, realizado el diseño de la solución al problema planteado y finalmente la implementación de dicho diseño, podemos afirmar que se han cumplido los objetivos del desarrollo de este proyecto de fin de grado para el cual se quería desarrollar un sistema de control de vuelo autónomo para un dron.

Se puede realizar esta afirmación, tras haber realizado un exhaustivo estudio de diferentes trayectorias y el comportamiento de nuestro sistema ante diferentes situaciones, realizando una comparación entre el comportamiento del sistema de vuelo autónomo implementado en nuestro dron conociendo sus limitaciones fruto de errores en la obtención de los datos de vuelo ya sean por los sensores del propio dron, o por errores producidos por factores externos al dron que afectan a la trayectoria que debe seguir ya que se ha observado que ante una misma trayectoria en determinadas situaciones el dron se comporta de diferente manera produciéndose diferentes valores de vuelo y localización, además de la aparición de pequeños errores en la estimación de la distancia y el tiempo en el que debería llegar el dron a un punto concreto debido al empleo de una velocidad de aceleración siempre constante que no varía dependiendo de la localización y la distancia a la que nos encontramos respecto del destino; frente al mismo sistema de vuelo autónomo empleando el sistema de control de vuelo que hemos implementado en nuestro software, el cual como se ha explicado durante la fase de diseño e implementación funciona simultáneamente al sistema de vuelo autónomo produciendo una serie de correcciones en cada instante, las cuales hemos visto que se realizan tanto para la velocidad, compara en todo momento la velocidad que lleva el dron frente a la que debería llevar y realiza una aceleración o deceleración en la fase de avance de nuestro sistema eliminando así la limitación de la velocidad siempre constante reduciendo considerablemente el error; como para la orientación que tiene el dron respecto al punto de destino, realizándose giros a la par que avanza el dron y se produce la modificación de la velocidad.

Como resultado de todo esto se ha demostrado tener un sistema que es capaz de realizar una trayectoria de notable complejidad de forma autónoma ajustándose de forma muy eficaz a la trayectoria ideal que debería seguir el dron, realizando las correcciones anteriormente mencionadas, que nos permite tener un buen control sobre el dron y que consiga llegar de forma satisfactoria al punto destino con el menor error de posición posible.

Además, se ha conseguido demostrar la validez de la navegación basada en sensores inerciales, y se abre la puerta a futuras extensiones que integren estos sensores con GPS para una alta fiabilidad del sistema ante caídas de la señal GPS. Esto se ha logrado tan

solo empleando un sistema de referencia local por el que el dron siempre es el origen y punto de referencia evitando así problemas de cobertura GPS.

Otro objetivo muy beneficioso para labores de investigación y navegación es que nuestro sistema es capaz de extraer en todo momento información de vuelo, la cual nos permite estudiar los cambios que se producen en un vuelo ante las situaciones posibles y poder actuar ante ellas de forma autónoma de forma similar a un sistema de inteligencia artificial no supervisado.

Por otro lado, se ha realizado el estudio de un nuevo sistema de vuelo autónomo que emplea un software libre y un protocolo de comunicaciones que está empezando a extenderse en la actualidad en la comunicación de vehículos no tripulados como es el software QgroundControl y el protocolo MavLink. De este último hemos encontrado multitud de utilidades y ventajas, como es una comunicación segura entre la base y el dron y la extracción de datos en tiempo real permitiéndonos manipularlos y seleccionar que datos queremos obtener. Por parte del software QGroundControl, hemos conseguido realizar el estudio y análisis de trayectorias autónomas de forma sencilla empleando en este caso coordenadas GPS, que sirve como alternativa a nuestro sistema de vuelo autónomo.

No obstante, durante el desarrollo y el estudio del proyecto han ido surgiendo numerosos problemas que en una primera instancia parecían muy complejos, pero que se han ido subsanando con una gran labor de investigación e implementación, que ha conllevado a tomar soluciones que se ajustan y satisfacen las necesidades de nuestro proyecto, pero que a gran escala puede tener limitaciones. Por este motivo posiblemente el sistema de control implementado no sea el más eficiente posible y se puedan implementar en el futuro mejoras que hagan de él un sistema más completo y con una menor tasa de error.

Las limitaciones ante las que me he encontrado, incluye la adaptación del Software LSIDrone al UAV escogido para la ejecución del proyecto, este dron además tiene bastantes restricciones a la hora de poder acceder a su información de vuelo y la forma de obtenerla, por ello hemos tenido que recurrir a numerosos cálculos matemáticos, que nos ayudasen a obtener la información que necesitábamos en todo momento. Por otra parte, he necesitado familiarizarme con un entorno de programación nuevo para mí al igual que el lenguaje de programación escogido, que al final ha resultado ser un gran acierto debido a sus ventajas. Otra limitación que se preveía era la de generalizar a otras marcas y modelos de drones dicho sistema, pero esta limitación no es tal ya que a pesar de que el Software LSIDrone sí que es determinado para nuestro dron, el código es fácilmente adaptable a otros softwares teniendo así la posibilidad de implementar este sistema de control de vuelo autónomo en otras plataformas.

Para concluir, todas estas limitaciones han servido para desarrollar un buen sistema de control, aprender el funcionamiento de un dron y mejorar en el control de los mismos, así como avanzar en la investigación sobre los sistemas de control actuales, los sistemas de posicionamiento y tener una experiencia de aeronavegación con el Parrot AR Drone 2.0 permitiéndome conocer un ámbito en plena expansión o en pleno auge como es el de los sistemas UAV, viendo multitud de utilidades nuevas que desconocía, así como un gran número de posibilidades con las que trabajar en un futuro inmediato. Por lo tanto, espero que el desarrollo de este trabajo de fin de grado haya resultado tan interesante para el lector, como para mí, el desarrollador, que tanto he aprendido y disfrutado realizándolo.

Tabla 10: Objetivos Cumplidos

TABLA DE OBJETIVOS CUMPLIDOS
Objetivos Principales
Investigación y análisis de técnicas de vuelo autónomo, sistemas UAV y de referencia.
Implantación de un sistema de vuelo autónomo sobre cuadricóptero.
Extracción y análisis de datos de vuelo.
Implantación de un sistema de control de vuelo autónomo sobre cuadricóptero
Estudio de viabilidad del sistema implantado: eficiencia de sus resultados
Objetivos secundarios
Estudio del protocolo de comunicaciones seguro para vehículos aéreos no tripulados: MavLink
Estudio e implantación de un sistema de vuelo autónomo para cuadricóptero: QgroundControl combinado con MavLink

10.2 VENTAJAS ECONÓMICAS PARA SU DISTRIBUCIÓN

Este proyecto se ha desarrollado como **software libre,** que como se sabe presenta numerosas ventajas sobre todo en el apartado económico para las grandes y pequeñas empresas que se interesen en este software para incluir en sus proyectos o en sus infraestructuras sin ver mermados sus intentos de crecimiento principalmente por el bajo coste en infraestructura y el inexistente coste en licencias. Otra de las características de ser un software libre es la posibilidad de ser instalado tantas veces como se requiera por parte del usuario sin costes añadidos, permitiendo entonces una **libertad de uso y redistribución.**

Además, al tratarse de un **software cerrado** no será necesario un nuevo evolutivo de trabajo (a petición del cliente) y tan solo un mantenimiento si así se desea, por lo que no es necesaria una futura inversión en este software.

Es un proyecto **independiente de la tecnología** ya que como se mencionó en el documento es compatible con cualquier sistema, por lo que no es necesario invertir en un software específico pudiendo abaratar costes.

Todas estas ventajas económicas hacen de este software, un software económico y flexible que seguro puede ser de gran interés para diferentes clientes.

Más adelante veremos el presupuesto de forma detallada.

10.3 TRABAJOS FUTUROS

Como he mencionado en repetidas ocasiones los sistemas aéreos no tripulados están en pleno auge en nuestra sociedad y se están empleando en multitud de campos, además tras haber realizado este trabajo de fin de grado he conseguido comprobar todo su potencial y entender cómo funcionan. Por ello veo en este ámbito una muy buena oportunidad tanto a nivel personal para seguir investigando y aprendiendo, como a nivel profesional puesto que considero que aún queda mucho por explotar en este ámbito y que debido a este auge existen muchas oportunidades laborales y de negocio.

Por lo tanto, a partir de este presente proyecto, tengo como objetivo realizar nuevos proyectos.

Empleando como base este sistema desarrollado, se podría realizar un proyecto que mejorase el sistema de control empleando un sistema de inteligencia artificial que fuese aprendiendo a través de una experiencia previa de comportamiento y datos extraídos, en este caso, todas las trayectorias realizadas en este proyecto y su consiguiente análisis. El sistema de control basado en inteligencia artificial, empleando una red de neuronas en concreto la idea sería utilizar un perceptrón multicapa, recibiría como entrada la diferencia entre las coordenadas reales e ideales y la velocidad, consiguiendo clasificar mediante el valor de su salida el giro de orientación que debe realizar. Esta red sería necesario entrenarla previamente mediante el conjunto de test, que serían las instancia como ya he mencionado, de este proyecto, y una vez se finalizase este proceso de entrenamiento obtendríamos la red de neuronas con su configuración que para una nueva instancia de vuelo con los datos extraídos obtener que giro debe realizar.

Otros posibles trabajos futuros a realizar serían incorporando la cámara del dron. Los proyectos que podrían realizarse serían principalmente emplear un algoritmo de reconocimiento de imágenes para detectar personas, o para detectar edificios u objetos que se interpongan en una trayectoria para poder evitarlos.

Por último, tras haber realizado un estudio e implantación en este dron del sistema de trayectorias autónomas QgroundControl y el protocolo de comunicaciones MavLink, sería interesar ahondar en estas tecnologías y continuar con la investigación para poder realizar nuevas aplicaciones, como por ejemplo realizar un sistema de control para QgroundControl.

11 CONCLUSIONS

After performing the problem analysis, studied its historical context and environment, make the design of the solution to the problem and finally the implementation of the design, we can say that they have met the objectives of the development of this end of degree project for which it wanted to develop a control system of autonomous flight for a drone.

I can make this claim, after making a comprehensive study of different paths and the behavior of our system in different situations, doing a comparison between the behavior of autonomous flight system implemented in our drone knowing their limitations result of errors in obtaining flight data either by itself drone sensors, or errors caused by external factors affecting the drone trajectory to be followed as has been observed that with a same path in certain situations the drone behaves differently producing different flight values and location, besides the appearance of small errors in the estimation of the distance and time that the drone should reach a certain point due to the use of an acceleration speed always constant that doesn't vary depending the location and the distance to the destination.

Compared to the same system of autonomous flight control that we have implemented in our software, which is explained during the design and implementation phase, it functions simultaneously to the autonomous flight system producing a series of correction in every moment, which we have seen are performed to the speed, it compares in every moment the drone speed against which should lead and performs acceleration or deceleration in the advance phase of our system thus eliminating the speed limit always significantly reducing the error constant; and for having the drone guidance regarding destination point, performing modification twists on the same moment that the drone are advancing and changing the speed it occurs.

The result of all of this has been shown to have a system that can to doing a remarkable complexity path of autonomously effectively being adjusted to the ideal trajectory that should follow the drone, making the corrections last mentioned, that allow us have a good control over the drone and get arrive correctly at the destination with the smallest error.

It has also succeeded in demonstrating the validity of the inertial sensors based navigation, and opens the door to future extensions that integrate these sensors with GPS for high reliability of the system to the GPS signal drops. This has been achieved only using a local reference system by which the drone is always the origin and reference point thus avoiding problems of GPS coverage.

Another very beneficial target for research work and navigation is that our system is able to extract all the time flight information, which allows us to study the changes that occur on a flight to possible situations and to act before them autonomously similarly to an artificial intelligence system unsupervised.

On the other hand, it has made the study of a new system of autonomous flight employing a free software and a communications protocol that is starting to spread today in communication unmanned vehicles such as QgroundControl software and MavLink protocol. The latter have found many utilities and advantages, such as secure communication between the base and the drone and extraction of data in real time allowing manipulate and select which data we get. QGroundControl by the software, we managed to conduct the study and analysis of autonomous paths easily in this case using GPS coordinates, which serves as an alternative to our system of autonomous flight.

However, during the development and study of the Project have appeared very problems that in a first instance seemed very complex but they have been overcome with a great work of investigation and implementation, that have led to take solutions that fit take and meet the needs of our project, but complications that can leads to large scale.

For this reason, possibly the implemented control system isn't the most efficient and could be implement in the future improvements that make it a more complete system with lower error rate.

The limitations to which I have found, including the adaptation of the Software LSIDrone for UAV chosen for the implementation of the project, this drone also has several restrictions when you can access your flight information and how to obtain it, so we have had to resort to numerous mathematical calculations, that would help to get the information we needed at all times. Moreover, I needed to familiarize myself with new programming environment for me as the chosen programming language, which ultimately proved to be a great success because of its advantages. Another limitation that was expected was to generalize to other makes and models of drones that system, but this limitation is not so because even though the Software LSIDrone yes that is given to our drone, the code is easily adaptable to other software thus having the possibility of implementing this system of autonomous flight control on other platforms.

To conclude, all these limitations have served to develop a good control system, learn the operation of a drone and improve control thereof, and advance research on current

control systems, positioning systems and have an experience of air navigation with the Parrot AR Drone 2.0 allowing me to know an area expanding or booming as that of the UAV systems, seeing many new utilities unknown, as well as a large number of possibilities that work on an immediate future. Therefore, I hope that the development of this work has been so interesting to the reader, as for me, the developer, who both have learned and enjoyed performing it.

Tabla 11: Goals Met

Goals Met Table
Main Objectives
Analysis and research of flight autonomous techniques, UAVs systems and reference systems.
Autonomous Flight System development over quadcopter UAV.
Extraction and analysis of data flying
Implementation of a control system of autonomous flight on quadcopter UAV
Viability study of the implemented system: efficiency of their results
Secondaries Objectives
Study of the protocol for secure communications UAVs: MavLink
Study and implementation of a system of autonomous flight for quadcopter: QgroundControl combined with MavLink

11.1 FUTURE WORK

As I mentioned repeatedly UAVs are booming in our society and are being used in many fields, besides having done this work EOG I managed to check their full potential and understand how they work. Therefore, I see in this area a very good opportunity both personally and for further research and learning, and professionally since I believe that much remains to be exploited in this area and because of this boom there are many job opportunities and business.

Therefore, from this present project it aims to carry out new projects.

Using this system developed, could make a project that would improve the control system using an artificial intelligence system that was learning through a prior extracted experience behavior and data, in this case, all paths carried out in this project and its subsequent analysis. The control system based on artificial intelligence, using a network of neurons in particular the idea would be to use a multilayer perceptron, receive as input the difference between the actual coordinates and ideals and speed, getting classified by the value of its output rotating guidance to be performed. This network would be necessary first to train through the whole test, which would be the instance as I mentioned, this project, and once finalize this training process would obtain the network of neurons with its configuration for a new instance of flight data extracted get that rotation should be performed.

Other possible future work to be done would be incorporating camera drone. Projects that could be made would mainly employ image recognition algorithm to detect people or buildings or to detect objects passing on a path to avoid them.

Finally, after conducting a study and implementation in this drone system of autonomous paths QgroundControl and communications protocol MavLink, it would be interested in continuing the study and research of these technologies to make new applications, such as performing a control system for QgroundControl.

12 GESTIÓN DEL PROYECTO Y PRESUPUESTO

12.1 PLANIFICACIÓN DEL PROYECTO

En este apartado se va a mostrar la planificación realizada al inicio del proyecto mediante un Gantt, con la descripción de cada una de las fases que lo forman así como el tiempo planificado para cada tarea. A continuación se mostrará el diagrama real con las tareas que se han llevado a cabo para el desarrollo final del proyecto y el tiempo que se ha necesitado para la realización de las mismas.

12.1.1 PLANIFICACIÓN INICIAL DEL PROYECTO

En la siguiente imagen se muestran las tareas que componen la planificación junto a los tiempos para llevarlas a cabo:

Nombre	Fecha de inicio	Fecha de fin
Instalación LSIDrone	1/01/16	31/01/16
Compilación en PC	1/01/16	8/01/16
Estudio del software	9/01/16	15/01/16
Conexión Dron-Software	16/01/16	19/01/16
Estudio posibilidades LSIDrone	20/01/16	31/01/16
Interfaz LSIDrone	20/01/16	30/01/16
Estudio del ámbito de los drones	1/01/16	14/02/16
Estudio de sistemas de control existentes	1/01/16	1/02/16
Estudio del funcionamiento del dron	1/02/16	14/02/16
Diseño del sistema de control	15/02/16	31/03/16
Estudio del problema	15/02/16	29/02/16
Diseño del problema	1/03/16	31/03/16
Implementación del sistema de control	1/04/16	29/05/16
Implementación	1/04/16	29/05/16
Refinamiento del sistema de Control	1/05/16	29/05/16
Análisis de resultados	1/05/16	29/05/16
Estudio de los resultados	1/05/16	29/05/16
Pruebas del sistema de control	15/05/16	29/05/16
PostMortem	1/01/16	14/07/16
Redacción del documento final	1/01/16	15/06/16
Preparación y elaboración de la defensa del proyecto	15/06/16	14/07/16

Ilustración 64: Tareas Planificación Inicial

A continuación, se muestra el diagrama de Gantt resultado:

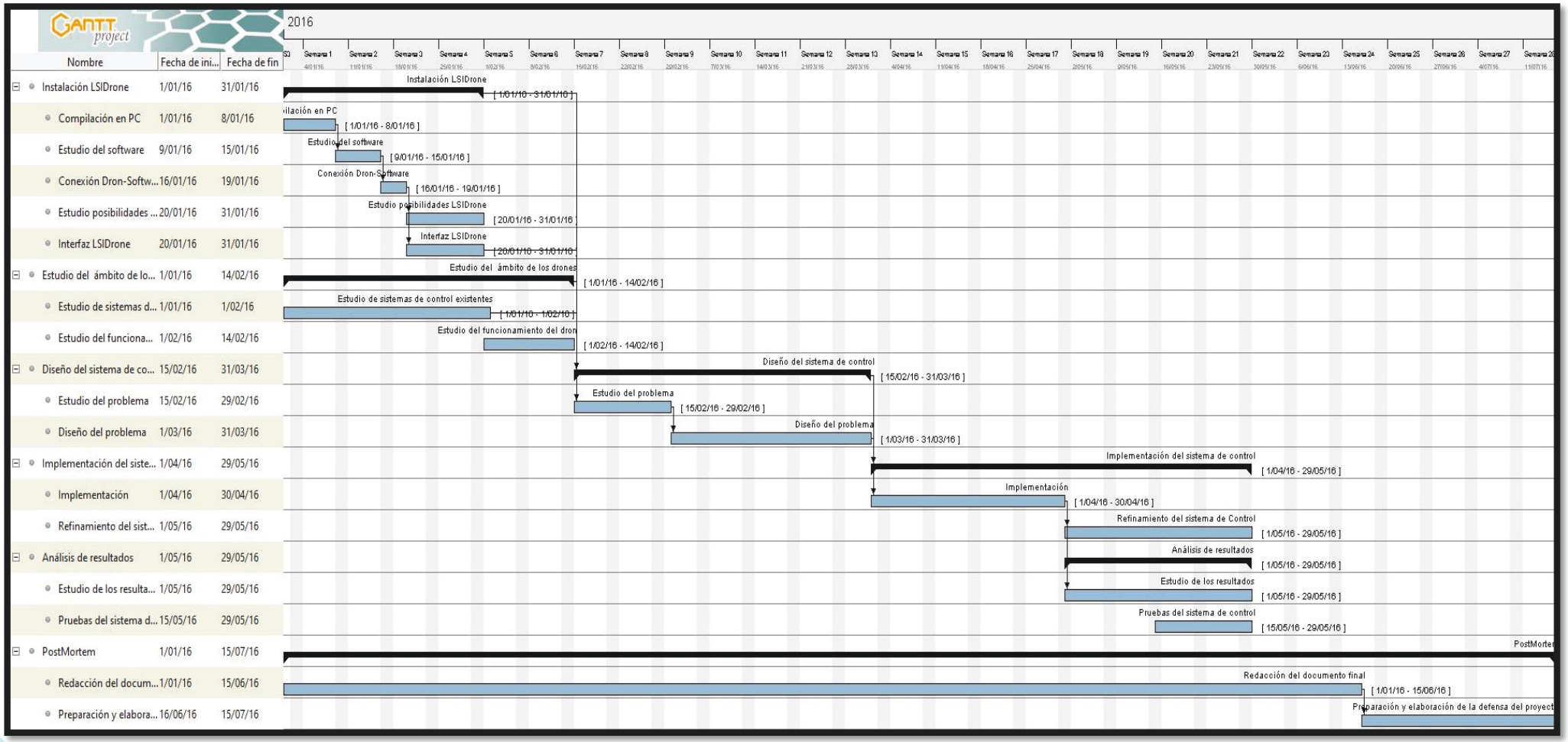


Ilustración 65:Diagrama de Gantt Planificación Inicial

12.1.1.1 DESCRIPCIÓN DE LAS TAREAS

a) Instalación LSIDrone.

Engloba todas las tareas necesarias para llevar a cabo la instalación del software LSIDrone.

Compilación en PC.

Esta tarea se trata de conseguir compilar y ejecutar el software en el ordenador empleado para el desarrollo del proyecto necesitando diferentes programas y complementos como son Visual Studio, SDK y las librerías necesarias.

Estudio del Software.

Esta tarea consiste en analizar y comprender el funcionamiento de las diferentes clases que forman el proyecto LSIDrone.

Conexión Dron-Software

Esta tarea consiste en realizar la conexión entre el dron, el ordenador y el software LSIDrone.

Estudio posibilidades LSIDrone.

Trata de estudiar y analizar las limitaciones del software y las utilidades que nos ofrece para poder desarrollar nuestra solución.

Interfaz LSIDrone.

Consiste en el desarrollo y despliegue de la interfaz gráfica.

b) Estudio del ámbito de los drones.

Esta fase conlleva el estudio de los diferentes entornos de control existentes, así como la familiarización con los diferentes drones, los conceptos principales y las posibilidades que nos ofrecen los vehículos no tripulados. En resumen, consiste en el estudio del marco teórico del proyecto.

c) Diseño del sistema de control

Estudio del problema.

Esta tarea consiste en determinar el problema que queremos resolver, y los objetivos que se quieren cumplir.

Diseño del problema

En esta fase se realiza el diseño de la solución técnica, los requisitos del problema a tratar, los caminos escogidos para resolver el problema y cómo queremos que funcione nuestro sistema.

d) Implementación del sistema de control.

Implementación.

Fase de desarrollo del código del diseño realizado en c#.

Refinamiento del sistema de control.

Consiste en optimizar el sistema de control implementado, realizando las modificaciones en la implementación pertinentes.

e) Análisis de resultados.

Estudio de los resultados y pruebas del sistema de control.

En esta fase, se realizan las pruebas y los análisis de las trayectorias de vuelo que realiza nuestro sistema implementado, así como la tasa de error y el estudio de viabilidad del sistema. Esta tarea se realiza conjuntamente en la fase de refinamiento del sistema de control, realizando numerosas pruebas y estudios para obtener la implementación óptima.

f) PostMortem.

Redacción del documento final.

Esta tarea se realiza en conjunto con el resto de tareas que consiste en redactar y documentar el trabajo realizado.

Preparación y elaboración de la defensa del proyecto.

Consiste en la preparación de las diapositivas de la presentación del proyecto, así como la preparación de la defensa.

12.1.2 PLANIFICACIÓN FINAL DEL PROYECTO

Nombre	Fecha de inicio	Fecha de fin
Instalación LSIDrone	1/01/16	20/02/16
Compilación en PC	1/01/16	15/01/16
Estudio del software	16/01/16	29/01/16
Conexión Dron-Software	30/01/16	5/02/16
Estudio posibilidades LSIDrone	6/02/16	20/02/16
Interfaz LSIDrone	6/02/16	20/02/16
Estudio del ámbito de los drones	1/01/16	4/03/16
Estudio de sistemas de control existentes	1/01/16	31/01/16
Estudio del funcionamiento del dron	1/02/16	21/02/16
Estudio de QGroundControl	21/02/16	4/03/16
Estudio protocolo MavLink	22/02/16	4/03/16
Diseño del sistema de Control	5/03/16	6/04/16
Estudio del problema	5/03/16	12/03/16
Diseño del problema	13/03/16	6/04/16
Diseño Sistema autónomo	13/03/16	23/03/16
Diseño sistema de extracción	24/03/16	31/03/16
Diseño sistema de Control	1/04/16	6/04/16
Implementación Sistema Control	7/04/16	26/06/16
Implementación	7/04/16	18/05/16
Implementación temporizadores	7/04/16	11/04/16
Implementación sistema autónomo	12/04/16	20/04/16
Implementación extracción de datos	21/04/16	4/05/16
Implementación sistema de control	5/05/16	18/05/16
Refinamiento del sistema de Control	19/05/16	26/06/16

Ejecución QGroundControl	5/03/16	26/04/16
Creación cabeceras MAvLink	5/03/16	18/03/16
Compilación QGroundControl	19/03/16	18/04/16
Inclusión cabeceras MavLink en QGround	19/04/16	22/04/16
Ejecución del software	23/04/16	26/04/16
Análisis de resultados Sistema de Control	19/05/16	26/06/16
Extracción de datos	19/05/16	25/05/16
Estudio de los resultados	26/05/16	26/06/16
Estudio y análisis gráfico	26/05/16	26/06/16
Análisis de errores	26/05/16	26/06/16
Pruebas del sistema de control	26/05/16	26/06/16
Realización de trayectorias Sistema de co...	26/05/16	26/06/16
Análisis sistema QGroundControl	27/04/16	25/05/16
Análisis de resultados	27/04/16	25/05/16
Ejecución de pruebas de vuelo	27/04/16	25/05/16
PostMortem	1/01/16	15/09/16
Redacción del documento final	1/01/16	15/09/16

Ilustración 66:Tareas Planificación Final

A continuación, se muestra el diagrama de Gantt resultado:

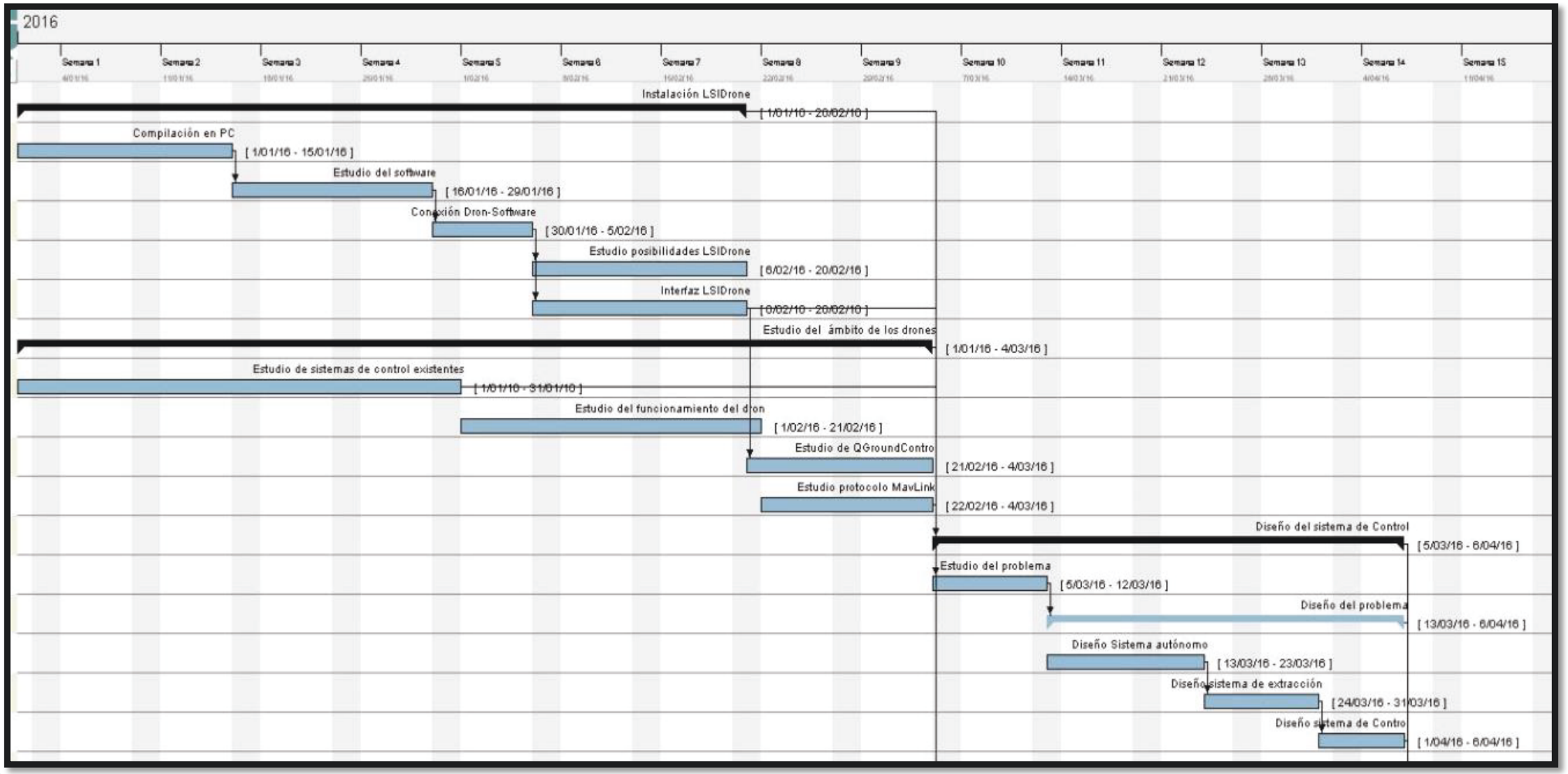


Ilustración 67: Gantt Planificación Final parte 1

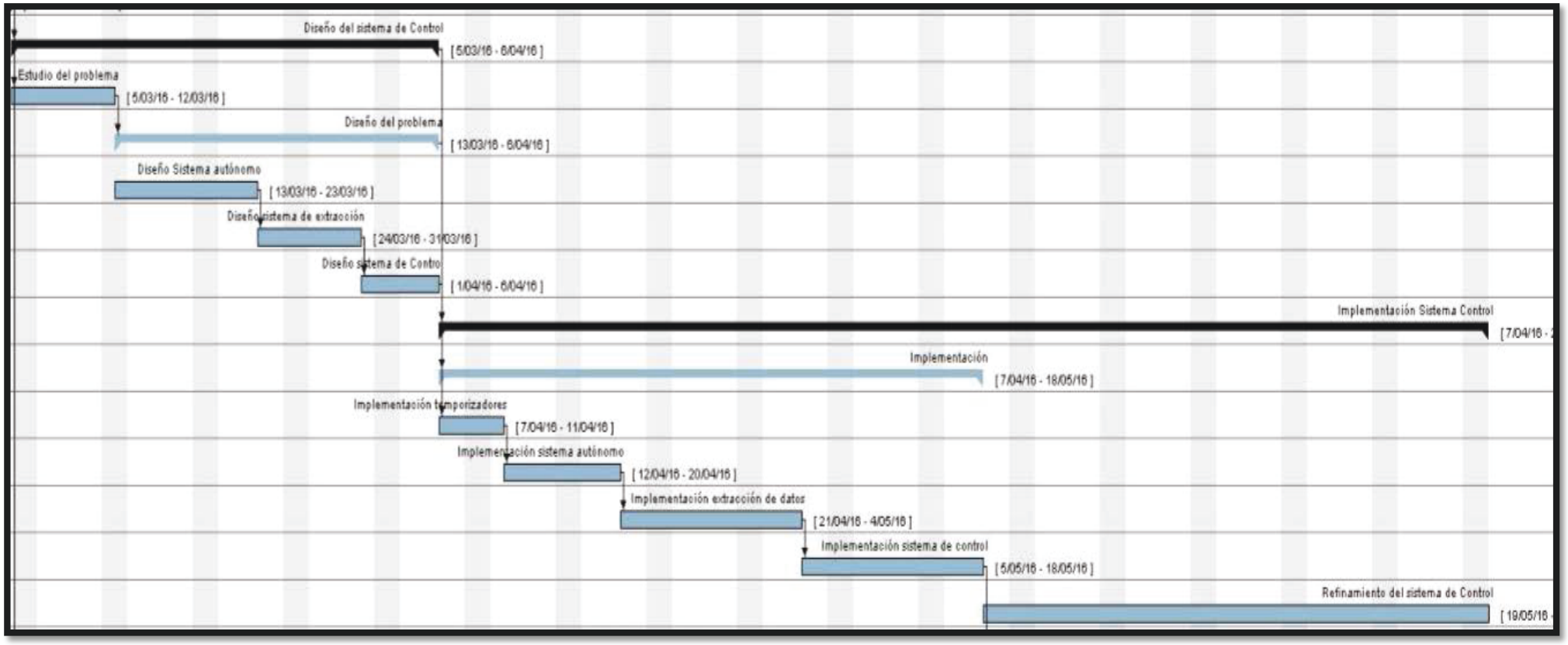


Ilustración 68:Gantt Planificación Final parte

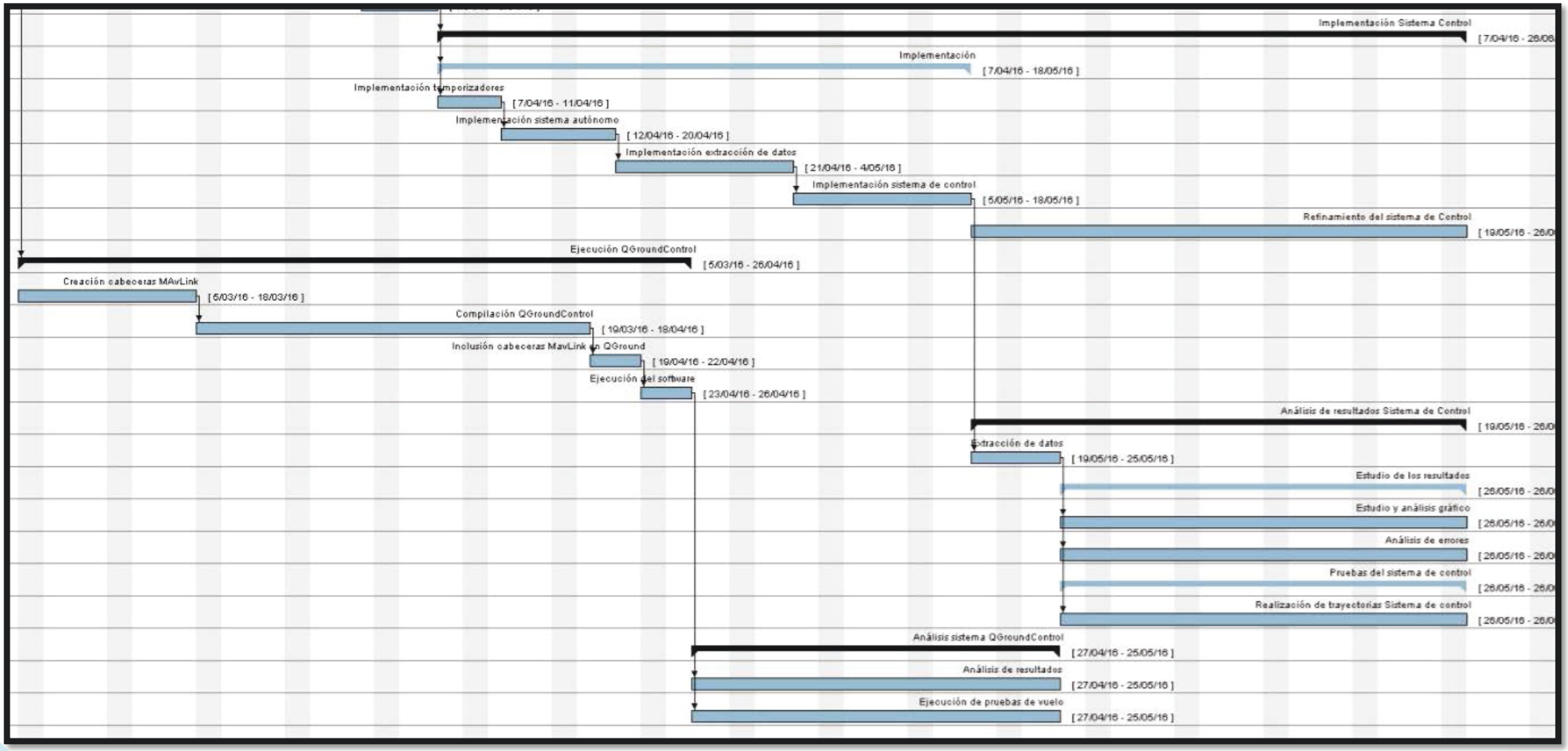


Ilustración 69: Gantt Planificación Final parte 3

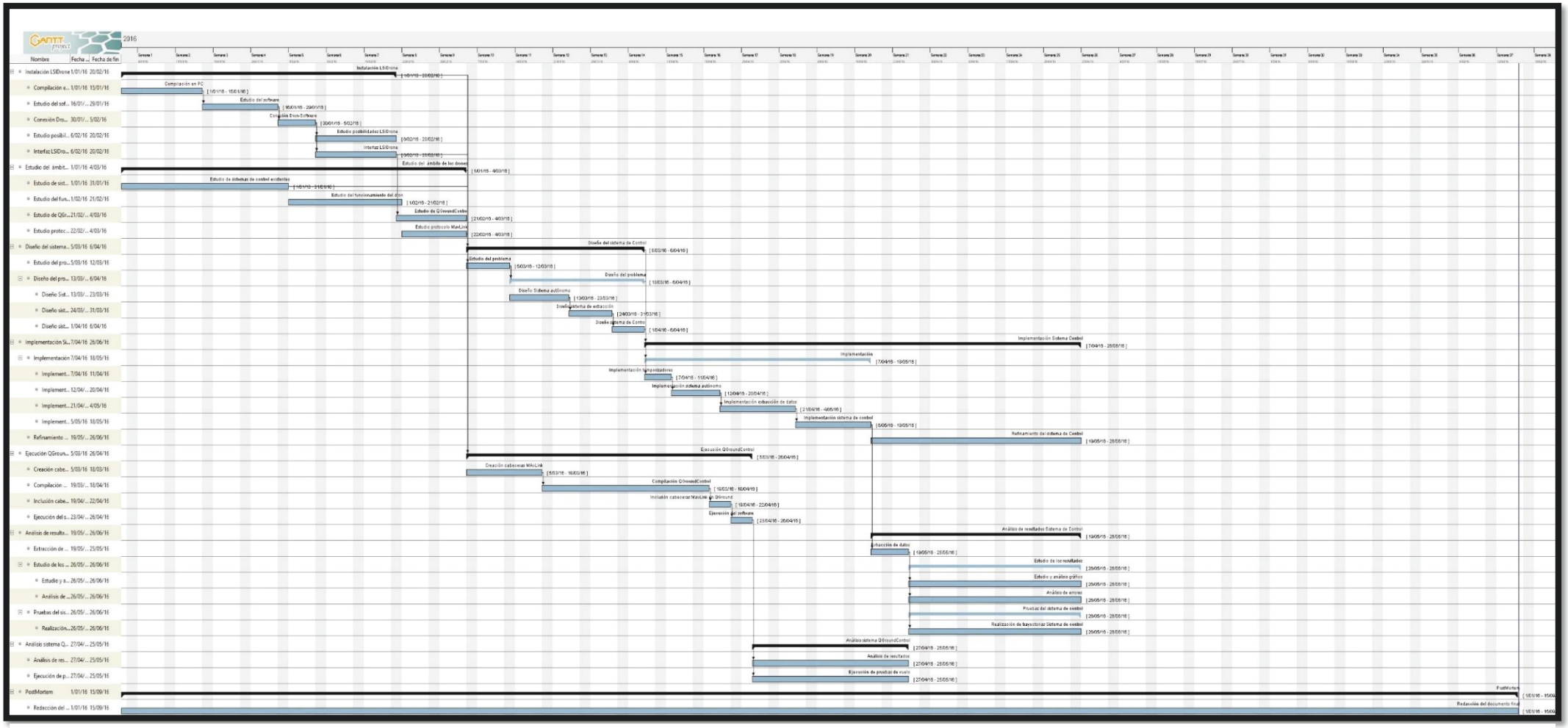


Ilustración 70: Diagrama de gantt Final

Las tareas que aparecen en el diagrama final de Gantt, son todas aquellas que se han realizado y explicado paso a paso a lo largo de todo el documento, hasta llegar a la versión final del proyecto que está redactado en este documento.

7.3 Instalación LSIDrone.

Engloba todas las tareas necesarias para llevar a cabo la instalación del software LSIDrone.

Compilación en PC.

Esta tarea se trata de conseguir compilar y ejecutar el software en el ordenador empleado para el desarrollo del proyecto necesitando diferentes programas y complementos como son Visual Studio, SDK y las librerías necesarias.

Estudio del Software.

Esta tarea consistió en analizar y comprender el funcionamiento de las diferentes clases que forman el proyecto LSIDrone.

Conexión Dron-Software

Esta tarea consistió en realizar la conexión entre el dron, el ordenador y el software LSIDrone.

Estudio posibilidades LSIDrone.

Se trató de estudiar y analizar las limitaciones del software y las utilidades que nos ofrece para poder desarrollar nuestra solución.

Interfaz LSIDrone.

Consiste en el desarrollo y despliegue de la interfaz gráfica.

7.4 Estudio del ámbito de los drones.

Esta fase conlleva el estudio de los diferentes entornos de control existentes, así como la familiarización con los diferentes drones, los conceptos principales y las posibilidades que nos ofrecen los vehículos no tripulados. En resumen, consiste en el estudio del marco teórico del proyecto.

Estudio QgroundControl

Tras haber realizado una fase previa de estudio de sistemas de control y vuelo existentes, se descubrió la utilización en expansión de una nueva tecnología QgroundControl para planificación de trayectorias, en esta fase se realizó su estudio y análisis de funcionamiento y posibilidad de implementación en nuestro dron.

Estudio MavLink

Esta fase se realizó de forma simultánea al estudio de QgroundControl, y consiste en el análisis de este nuevo protocolo, sus características y posibilidades.

7.5 Diseño del sistema de control

Estudio del problema.

Esta tarea consistió en determinar el problema que queríamos resolver, y los objetivos que se querías cumplir, así como las posibilidades para abordar dicho problema.

Diseño del problema

En esta fase se realizó el diseño de la solución técnica, los requisitos del problema y su posible implementación. Finalmente se dividió en tres sistemas la solución a este proyecto que funcionarían de forma conjunta formando tres fases de diseño diferenciadas.

-**Diseño Sistema Autónomo**

-Diseño Sistema de Extracción

-Diseño Sistema de Control

7.6 Implementación del sistema de control.

Implementación.

Fase de desarrollo del código del diseño realizado en c# del diseño realizado previamente. La implementación se realizó en cuatro fases, cuyo objetivo se ha detallado en la memoria.

-**Implementación de temporizadores.**

-Implementación del sistema autónomo.

-Implementación del sistema de extracción de datos.

-Implementación del sistema de control.

Refinamiento del sistema de control.

Consiste en optimizar el sistema de control implementado, realizando las modificaciones en la implementación pertinentes.

7.7 Ejecución QGroundControl

Esta fase engloba todas aquellas tareas que fueron necesarias para la compilación y puesta en marcha del sistema previamente estudiado para vuelo autónomo del dron en mi ordenador.

7.8 Análisis de resultados Sistema de Control.

Extracción de datos

Esta tarea consistió en realizar diferentes extracciones de datos durante una serie de trayectorias para poder ser analizadas posteriormente. Esta tarea se realizó posteriormente a la implementación del sistema de control.

Estudio de los resultados y Pruebas del sistema de Control.

En esta fase, se realizaron las tareas que engloban las pruebas y los análisis de las trayectorias de vuelo a partir de los datos extraídos por el sistema, así como la tasa de error y el estudio de viabilidad del sistema. Esta tarea se realiza conjuntamente en la fase de refinamiento del sistema de control, realizando numerosas pruebas y estudios para obtener la implementación óptima.

7.9 Análisis sistema QGroundControl.

Tras haber realizado la fase de ejecución del software, se inició esta nueva fase que conllevaba a un estudio de las diferentes posibilidades de este software, así como la ejecución de diferentes trayectorias.

7.10 PostMortem.

Redacción del documento final.

Esta tarea se realiza en conjunto con el resto de tareas que consiste en redactar y documentar el trabajo realizado.

7.11 PRESUPUESTO

Tras haber visto las diferentes ventajas y características que afectan en el presupuesto de este proyecto para su distribución y venta (ver Ventajas económicas para su distribución), además de la planificación y tiempo empleado para el desarrollo del proyecto se puede desglosar el presupuesto de este proyecto:

Tabla 12:Recursos Humanos

Recursos Humanos		
Duración del Proyecto	Días	259
	Semanas	37
	Horas/Día	6
	Total de Horas	1.554
Mano de obra	12 €/hora	

Tabla 13: Hardware

Hardware		
Equipo Informático (si se desea)	Ordenador Gama Media	300 €
Dron (si se desea)	AR Drone 2.0	289 €

Tabla 14:Software

Software	
Compilador Microsoft Visual Studio Ultimate 2012	0 €
LSIDrone	0 €
Libre Office	0 €

A continuación, se han elaborado dos presupuestos, a partir del desglose anterior dependiendo de si el cliente decide incorporar el hardware en el presupuesto o no:

PRESUPUESTO 1/2016

Universidad Carlos III	*Datos cliente*
Dirección: Campus de Colmenarejo	***Nombre y apellidos: Jesús García Herrero***
Población: Colmenarejo	***Provincia: Madrid***
Provincia: Madrid	
CIF / NIF: N/A	

Fecha de presupuesto:	*26-09-16*	***Validez:***	*365*	*días*	

DESCRIPCION	UNIDADES	PRECIO	% DTO.	PRECIO DTO.	TOTAL
Recursos Humanos	1.554	10,00	21%	7,90	12.276,60
Equipo Informático	1	300,00	10%	270,00	270,00
Drone	1	289,00	10%	260,10	260,10
LSIDrone	1	0,00	0%	0,00	0,00
Compilador	1	0,00	0%	0,00	0,00
			TOTAL BRUTO		**12.806,70**
		I.V.A. %		**21%**	**2.689,41**
	Total presupuesto …				15.496,11 €

Forma de pago:	**Cheque/ingreso en cuenta/ en metálico (lo que corresponda)**
Datos y Firma persona/empresa que confecciona el presupuesto.	ACEPTO EL PRESUPUESTO. Nombre, apellidos y firma del cliente.

Tabla 15:Presupuesto 1 Hardware incluido

PRESUPUESTO 2/2016

Universidad Carlos III		*Datos cliente*				
Dirección: Campus de Colmenarejo		***Nombre y apellidos: Jesús García Herrero***				
Población: Colmenarejo		***Provincia: Madrid***				
Provincia: Madrid						
CIF / NIF: N/A						
Fecha de presupuesto:	*26-09-16*	***Validez:***	*365*	*días*		

DESCRIPCION	UNIDADES	PRECIO	% DTO.	PRECIO DTO.	TOTAL
Recursos Humanos	1.554	10,00	21%	7,90	12.276,60
Equipo Informático	0	300,00	10%	270,00	0,00
Drone	0	289,00	10%	260,10	0,00
LSIDrone	1	0,00	0%	0,00	0,00
Compilador	1	0,00	0%	0,00	0,00
		TOTAL BRUTO			**12.276,60**
		I.V.A. %		**21%**	**2.578,09**
	Total presupuesto ...				14.854,69 €
Forma de pago:	**cheque/ingreso en cuenta/ en metálico (lo que corresponda)**				
Datos y Firma persona/empresa que confecciona el presupuesto.	ACEPTO EL PRESUPUESTO. Nombre, apellidos y firma del cliente.				

Tabla 16: Presupuesto 2 Hardware No incluido

BIBLIOGRAFÍA

ardronespain. (2016). *ardronespain*. Obtenido de http://www.ardronespain.com/sobre/flight-recorder/

Area Tecnología. (2015). Obtenido de http://www.areatecnologia.com/aparatos-electronicos/drones.html

Balasubramanian, S. (2016). *http://api.ning.com/*. Obtenido de http://api.ning.com/files/i*tFWQTF2R*7Mmw7hksAU-u9IABKNDO9apguOiSOCfvi2znk1tXhur0Bt00jTOldFvob-Sczg3*lDcgChG26QaHZpzEcISM5/MAVLINK_FOR_DUMMIESPart1_v.1.1.pdf

Crespo, G. (2014). *Sistema de enlace robusto para la teleoperación de un UAV (vehículo aéreo no tripulado) en la plataforma robótica ARGOS.*

Criado, R. M. (2014). *Diseño de estrategias de control para el seguimiento de trayectorias de un cuadricóptero comercial.* Sevilla.

Criado, R. M. (9 diciembre 2014). Diseño de estrategias de control para el seguimiento de trayectorias de un cuadricóptero comercial. En *Sistema de coordenadas y maniobras de vuelo.*

Díaz, S. M. (2016). *Diseño y Construcción de un Quadcopter.*

Dronespain. (2016). *Dronespain.com*. Obtenido de NUEVA LEY DE DRONES EN ESPAÑA – MARZO 2016: http://www.dronespain.pro/nueva-ley-de-drones-en-espana-marzo-2016/

Farjas, M. (Junio de 2016). *http://ocw.upm.es/ingenieria-cartografica-geodesica-y-fotogrametria/topografia-ii/Teoria_GPS_Tema_12.pdf*. Obtenido de Aplicaciones Topográficas del G.P.S: http://ocw.upm.es/ingenieria-cartografica-geodesica-y-fotogrametria/topografia-ii/Teoria_GPS_Tema_12.pdf

Fuentes Brea, J. P. (s.f.). Arquitectura cognitiva híbrida para la navegación autónoma de UAVS mediante mapas topológicos visuales.

Garrido, S. (4 de Julio de 2014). *Regulación de Drones en España*. Obtenido de http://language.hoganlovells.com/files/Publication/4700735a-c847-4597-914c-8012b7d4919b/Presentation/PublicationAttachment/d68f684e-f00e-43cf-b509-8201c7bb5d04/Legal%20flash%20regulaci%C3%B3n%20drones%20en%20Espa%C3%B1a.pdf

geográfico, I. (2016). *IGN*. Obtenido de Geodesia: https://www.ign.es/ign/layoutIn/actividadesGeodesiaStmagd.do

Gitbooks. (2016). *erlerobotics.gitbooks.io*. Obtenido de https://erlerobotics.gitbooks.io/erlerobot/content/es/mavlink/mavlink.html

Gómez, S. S. (2016). *mejordefiniciones.blogspot.com*. Obtenido de http://mejordefiniciones.blogspot.com.es/2015/06/que-es-un-dron-vehiculo-aereo-no.html

Hedrich, E. M. (2015). *Piloto de RPAS Cuadricoptero - Guía de referencia.* ISBN 978-84-606-5467-4.

Ley de regulación de drones en España 18/2014, 50 (15 de 10 de 2014).

Mínguez, G. F. (Abril,2009). *Integración Kalman de sensores inerciales INS con GPS en un UAV*. Obtenido de Upcommons: http://upcommons.upc.edu/bitstream/handle/2099.1/6930/memoriadef.pdf

Nacional, I. G. (2016). *ign*. Obtenido de http://www.ign.es/ign/main/index.do

Parrot. (Julio de 2016). *ardrone-2*. Obtenido de http://ardrone-2.es/especificaciones-ar-drone-2/

Parrot. (s.f.). ARDrone Developer Guide. En *Communication services between the AR.Drone 2.0 and a client.*

Parrot. (s.f.). ARDrone Developer Guide. En *AT Commands.*

Pose, C. (2015). Diseño de algoritmos de navegación y control para un hexarotor. En C. Pose, *Tipos de UAVS.* Argentina.

Poveda, C. M. (2012-2013). *Riunet*. Obtenido de https://riunet.upv.es/bitstream/handle/10251/32992/Memoria.pdf?sequence

QGroundControl. (2016). *qgroundcontrol.org*. Obtenido de http://qgroundcontrol.org/

qgroundcontrol.org. (Julio de 2016). Obtenido de http://qgroundcontrol.org/mavlink/start

Quirós, G. C. (2015-02-22). Sistema de enlace robusto para la teleoperación de un UAV (vehículo aéreo no tripulado) en la plataforma robótica ARGOS.

Rizo, D. R. (2014). *Planeación de trayectorias para un robot aéreo usando GPS.* Bogotá.

Upcommons. (junio de 2016). Obtenido de Diseño de un quadcopter, Funcionamiento y Movimiento de Quadcopter: https://upcommons.upc.edu/bitstream/handle/2099.1/21902/102664.pdf?sequence=1&isAllowed=y

Vicedo, V. R. (2013-2014). *Quadcopter basado en microcontrolador.*

Wikipedia. (2016). *Wikipedia*. Obtenido de https://es.wikipedia.org/wiki/Veh%C3%ADculo_a%C3%A9reo_no_tripulado#Clasificaci.C3.B3n_de_los_drones

Wikipedia. (20 de Julio de 2016). *Wikipedia.org*. Obtenido de https://en.wikipedia.org/wiki/MAVLink

ACRÓNIMOS Y DEFINICIONES

Tabla 17: Acrónimos y Definiciones

Nombre	Definición
UAV	Unmanned Aerial Vehicle también llamado dron es una aeronave que vuela sin tripulación.
Sistema de Control o FMS	Es un componente de la aviónica de una aeronave que permite automatizar una variedad de tareas a realizar durante un vuelo.
Sistema de vuelo autónomo	Sistema de vuelo no tripulado.
GPS	El sistema de posicionamiento global (GPS) es un sistema que permite determinar en toda la Tierra la posición de un objeto
Aeronave R/C	Aeronave de RadioControl
UCAV	Aviones no Tripulados de Combate
Logística	Conjunto de los medios necesarios para llevar a cabo un fin determinado de un proceso complicado.
Implementación	Una implementación es la instalación de una aplicación informática, realización o la ejecución de un plan, idea, modelo científico, diseño, especificación, estándar, algoritmo o política.
Pitch	Ángulo de navegación: Elevación.
Roll	Ángulo de navegación: Alabeo.
Yaw	Ángulo de navegación: Cabeceo.
NAVSTAR	Serie de 24 satélites de navegación que completan el Sistema de posicionamiento global.
WGS-84	El WGS84 es un sistema de coordenadas geográficas mundial que permite localizar cualquier punto de la Tierra.
ED-50	En España el ED50 ha sido el sistema oficial de la cartografía de la Península y Baleares hasta 2008.

Geoide	Forma teórica de la Tierra determinada por la geodesia en la cual se toma como superficie teórica el nivel medio de los mares.
Elipsoide de revolución	La superficie que se obtiene al girar una elipse alrededor de uno de sus ejes principales.
WayPoint	Los waypoints son coordenadas para ubicar puntos de referencia tridimensionales utilizados en la navegación basada en GPS.
Parrot	Parrot es el primer fabricante de dispositivos manos libres del mundo. Su sede central está en París, cuenta en la actualidad con 450 colaboradores en todo el mundo y genera el 85% de sus ventas en el exterior.
Interfaz	Dispositivo capaz de transformar las señales generadas por un aparato en señales comprensibles por otro. Dispositivo que conecta dos aparatos o circuitos
RGB	Sigla en inglés de red, green, blue, en español «rojo, verde y azul», es la composición del color en términos de la intensidad de los colores primarios de la luz.
Píxel	Un píxel o pixel, del plural píxeles (acrónimo del inglés picture element, 'elemento de imagen'), es la menor unidad homogénea en color que forma parte de una imagen digital.
HD	Siglas en inglés "High Definition", que hace referencia a la alta definición de un sistema de imagen o vídeo.
Giroscopio	Es un dispositivo mecánico que sirve para medir, mantener o cambiar la orientación en el espacio de algún aparato o vehículo.
USB	USB (Universal Serial Bus) es un tipo de dispositivo de almacenamiento de datos que utiliza memoria flash para guardar datos e información.
Wi-fi	El wifi (nombre común en español proveniente de la marca Wi-Fi)1 es un mecanismo de conexión de dispositivos electrónicos de forma inalámbrica.
SDK	Un kit de desarrollo de software o SDK (siglas en inglés de software development kit) es generalmente un conjunto de herramientas de desarrollo de software que le permite al programador o desarrollador de software crear aplicaciones para un sistema concreto.

RPM	Revoluciones Por Minuto
Opensource	Open Source o código abierto es el software distribuido y desarrollado libremente.
Dirección IP	Una dirección IP es un número que identifica, de manera lógica y jerárquica, a una Interfaz en red (elemento de comunicación/conexión) de un dispositivo (computadora, tableta, portátil, smartphone) que utilice el protocolo IP (Internet Protocol), que corresponde al nivel de red del modelo TCP/IP.
Servidor DHCP	DHCP (siglas en inglés de Dynamic Host Configuration Protocol, en español «protocolo de configuración dinámica de host») es un servidor que usa protocolo de red de tipo cliente/servidor en el que generalmente un servidor posee una lista de direcciones IP dinámicas y las va asignando a los clientes.
ACK	Un ACK (del inglés acknowledgement, en español acuse de recibo o asentimiento), en comunicaciones entre computadores, es un mensaje que el destino de la comunicación envía al origen de esta para confirmar la recepción de un mensaje.
Licencia LGPL	La licencia pública general limitada de GNU, o mayormente conocida por su nombre en inglés GNU Lesser General Public License, o simplemente por su acrónimo del inglés GNU LGPL es una licencia de software creada por la Free Software Foundation.